

```

UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTT  PPPPPPPPPPPP  SSSSSSSSSSSS  YYY      YYY
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTT  PPPPPPPPPPPP  SSSSSSSSSSSS  YYY      YYY
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTT  PPΓPPPPPPPPP  SSSSSSSSSSSS  YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEE                TTT          PPP      PPP  SSS      YYY      YYY
UUU      UUU  EEEEEEEEEEEEEEE  TTT          PPPPPPPPPPPPP  SSSSSSSSSS  YYY
UUU      UUU  EEEEEEEEEEEEEEE  TTT          PPPPPPPPPPPPP  SSSSSSSSSS  YYY
UUU      UUU  EEEEEEEEEEEEEEE  TTT          PPPPPPPPPPPPP  SSSSSSSSSS  YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUU      UUU  EEE                TTT          PPP          SSS      YYY
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT          PPP          SSSSSSSSSSSS  YYY
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT          PPP          SSSSSSSSSSSS  YYY
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT          PPP          SSSSSSSSSSSS  YYY

```

SA
VO[illegible]

(1)	57	DECLARATIONS
(1)	236	R/W PSECT
(1)	390	SATSSS05
(1)	439	SNDACC TESTS
(1)	514	SNDERR_S TESTS
(2)	565	SNDOPR TESTS
(2)	720	SNDSMB TESTS
(2)	980	REG_SAVE
(2)	1001	REG_CHECK
(2)	1043	PRINT_FAIL
(2)	1089	READ_CHECK
(2)	1129	CRE_JOB
(2)	1215	BUF_CHECK
(2)	1266	SND_CHECK
(2)	1306	GENREQ
(2)	1329	MODE_ID


```
0000 1 .TITLE SATSSS05 - SATS SYSTEM SERVICE TESTS (SUCC S.C.)
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24
0000 25 *****
0000 26
0000 27
0000 28
0000 29 ++
0000 30 FACILITY: SATS SYSTEM SERVICE TESTS
0000 31
0000 32 ABSTRACT: The SATSSS05 module tests the execution of the following
0000 33 VMS system services:
0000 34
0000 35 $SNDACC
0000 36 $SNDERR
0000 37 $SNDOPR
0000 38 $SND SMB
0000 39
0000 40
0000 41 ENVIRONMENT: User mode image.
0000 42 Needs CMKRNL privilege and dynamically acquires other
0000 43 privileges, as needed.
0000 44
0000 45 AUTHOR: Larry D. Jones, CREATION DATE: JULY, 1978
0000 46
0000 47 MODIFIED BY:
0000 48
0000 49 V03-002 PCG0001 Peter C. George 16-Feb-1981
0000 50 Add OPCMSG macro expansion
0000 51
0000 52 V03-001 LDJ0001 Larry D. Jones 17-Sep-1980
0000 53 Modified to conform to new build command procedures.
0000 54
0000 55 **
0000 56 --
```

```
0000 57 .SBTTL DECLARATIONS
0000 58 :
0000 59 : MACRO LIBRARY CALLS
0000 60 :
0000 61 $ACCDDEF ; accounting definitions
0000 62 $DIBDEF ; device info block offsets
0000 63 $EMBDEF ; error log buffer definitions
0000 64 $JBCMSGDEF ; job controller definitions
0000 65 $OPCDEF ; operator communications def.
0000 66 $OPCMMSG ; operator communications messages
0000 67 $OPRDEF ; operator message definitions
0000 68 $PHDDEF ; process header definitions
0000 69 $PRVDEF ; privilege definitions
0000 70 $SHR MESSAGES UETP,116,<<TEXT,INFO>> ; UETPS TEXT definition
0000 71 $SMRDEF ; symbiot manager definitions
0000 72 $STSDEF ; STS definitions
0000 73 $UETPDEF ; UETP message definitions
0000 74 :
0000 75 : Equated symbols
0000 76 :
00000000 0000 77 WARNING = 0 ; warning severity value for msgs
00000001 0000 78 SUCCESS = 1 ; success
00000002 0000 79 ERROR = 2 ; error
00000003 0000 80 INFO = 3 ; information
00000004 0000 81 SEVERE = 4 ; fatal
0000 82 :
0000000D 0000 83 CR = 13 ; terminal definitions
0000000A 0000 84 LF = 10
0000 85 :
00000006 0000 86 FIDSIZ = 6 ; ID sizes
00000006 0000 87 DIDSIZ = 6
00000014 0000 88 FILNAMSIZ = 20
00000007 0000 89 COM_FIL_SIZ = 7
0000 90 :
00000064 0000 91 BUF_SIZE=100 ; buffer size
0000 92 :
0000 93 ALL_OPR = OPCSM_NM_CENTRL!OPCSM_NM_PRINT!-
0000 94 OPCSM_NM_TAPES!OPCSM_NM_DISKS!-
0000 95 OPCSM_NM_DEVICE!OPCSM_NM_OPER1!-
0000 96 OPCSM_NM_OPER2!OPCSM_NM_OPER3!-
0000 97 OPCSM_NM_OPER4!OPCSM_NM_OPER5!-
0000 98 OPCSM_NM_OPER6!OPCSM_NM_OPER7!-
0000 99 OPCSM_NM_OPER8!OPCSM_NM_OPER9!-
0000 100 OPCSM_NM_OPER10!OPCSM_NM_OPER11!-
00FFF01F 0000 101 OPCSM_NM_OPER12
0000 102 :
0000 103 : ***** NOTE *****
0000 104 :
0000 105 : THE FOLLOWING DEFINITION IS TO BE REMOVED WHEN VMS RELEASE 2 IS FIXED.
0000 106 :
00000008 0000 107 SNDACCS_CHAN = 8
0000 108 : MACROS
0000 109 :
```



```
00000000 111 .PSECT RODATA,RD,NOWRT,NOEXE, LONG
0000 112
0000 113 TEST_MOD_NAME:
35 30 53 53 53 54 41 53 00' 0000 114 .ASCIC /SATSSS05/ ; needed for SATSMS message
08 0000
0009 115 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000' 0009 116 .ASCIC /SATSSS05/ ; module name
35 30 0017
0019 117 TEST_MOD_BEGIN:
6E 75 67 65 62 00' 0019 118 .ASCIC /begun/
05 0019
001F 119 TEST_MOD_SUCC:
6C 75 66 73 73 65 63 63 75 73 00' 001F 120 .ASCIC /successful/
0A 001F
002A 121 TEST_MOD_FAIL:
64 65 6C 69 61 66 00' 002A 122 .ASCIC /failed/
06 002A
0031 123 SNDACC:
43 43 41 44 4E 53 00' 0031 124 .ASCIC /SNDACC/
06 0031
0038 125 SNDERR:
52 52 45 44 4E 53 00' 0038 126 .ASCIC /SNDERR/
06 0038
003F 127 SNDOPR:
52 50 4F 44 4E 53 00' 003F 128 .ASCIC /SNDOPR/
06 003F
0046 129 SNDSMB:
42 4D 53 44 4E 53 00' 0046 130 .ASCIC /SNDSMB/
06 0046
004D 131 CS1:
21 20 74 73 65 54 00000055'010E0000' 004D 132 .ASCIC \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41 005B
70 65 74 73 20 43 41 21 20 65 6D 61 0067
2E 64 65 6C 69 61 66 20 4C 55 21 20 0073
007F 133 CS2:
74 63 65 70 78 45 00000087'010E0000' 007F 134 .ASCIC \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65 008D
41 21 20 64 65 76 69 65 63 65 72 20 0099
4C 58 21 20 3D 20 53 00A5
00AC 135 CS3:
74 63 65 70 78 45 000000B4'010E0000' 00AC 136 .ASCIC \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65 00BA
64 65 76 69 65 63 65 72 20 4C 58 21 00C6
58 21 20 3D 20 42 55 21 53 41 21 20 00D2
4C 00DE
00DF 137 CS5:
77 20 65 64 6F 4D 000000E7'010E0000' 00DF 138 .ASCIC \Mode was !AS.\
2E 53 41 21 20 73 61 00ED
00F4 139 CS6:
74 63 65 70 78 45 000000FC'010E0000' 00F4 140 .ASCIC \Expected byte offset !UB(10) = !XB(16) received !XB(16).\
73 66 66 6F 20 65 74 79 62 20 64 65 0102
3D 20 29 30 31 28 42 55 21 20 74 65 010E
63 65 72 20 29 36 31 28 42 58 21 20 011A
36 31 28 42 58 21 20 64 65 76 69 65 0126
2E 29 0132
0134 141 UM:
72 65 73 75 0000013C'010E0000' 0134 142 .ASCIC \user\
```

```
42 4D 24 54 53 53 00000148'010E0000' 0140 143 MBNAM:
58 0140 144 .ASCID \SST$MBX\
014F 145 TTNAM:
41 54 54 5F 00' 014F 146 .ASCIC \_TTA\ ; terminal name to send opr messages to
04 014F 147 TTUNIT:
0001 0154 148 .WORD 1 ; unit number for above
0156 149 EXP:
73 75 74 61 74 73 0000015E'010E0000' 0156 150 .ASCID \status\
0164 151 BAT_IMP_EXC:
20 68 63 74 61 42 0000016C'010E0000' 0164 152 .ASCID \Batch job improperly executed.\
72 65 70 6F 72 70 6D 69 20 62 6F 6A 0172
2E 64 65 74 75 63 65 78 65 20 79 6C 017E
018A 153 YES_DESC:
00000003 018A 154 .LONG 3
00000192' 018E 155 .ADDRESS SYM_NAME
0192 156 SYM_NAME: ; batch job symbol name
4D 59 53 0192 157 .ASCII \SYM\
0195 158 SYM_DESC:
00000014 0195 159 .LONG 20
0000038A' 0199 160 .ADDRESS SYM
019D 161 YES:
53 45 59 00' 019D 162 .ASCIC \YES\ ; parameter for SNDSMB
03 019D
01A1 163 QUENAM1:
55 51 5F 54 41 42 5F 50 54 45 55 00' 01A1 164 .ASCIC /UETP_BAT_QUE1/
31 45 01AD
0D 01A1
0000000E 01AF 165 QUENAM1L=-QUENAM1
01AF 166 QUENAM2:
55 51 5F 54 41 42 5F 50 54 45 55 00' 01AF 167 .ASCIC /UETP_BAT_QUE2/
32 45 01BB
0D 01AF
0000000E 01BD 168 QUENAM2L=-QUENAM2
01BD 169 MSGVEC:
00000003 01BD 170 .LONG 3 ; PUTMSG message vector
00741133 01C1 171 .LONG UETPS_TEXT
00000001 01C5 172 .LONG 1
00000169' 01C9 173 .ADDRESS MESSAGEL
01CD 174 TEST_ERROR: ; SNDERR test data
00000064 01CD 175 .LONG BUF_SIZE
000001D5' 01D1 176 .ADDRESS +4
00000000 01D5 177 A=0
01D5 178 .REPT BUF_SIZE
01D5 179 .BYTE A
01D5 180 A=A+1
00 01D5 181 .ENDR
0239 182 OPNAME:
41 50 4F 5F 00' 0239 183 .ASCIC /_OPA/ ; operator console name
04 0239
023E 184 OP_MSG1:
00000036' 023E 185 .LONG MSG1L ; GENREQ routine OPRMSG buffer
00000246' 0242 186 .ADDRESS +4
03 0246 187 .BYTE OPC$ RQ RQST
00000001 0247 188 .LONG OPC$M_NM_CENTRL ; request operator type
0000024A 024B 189 .=-1 ; is only 3 bytes big
```



```
00000000 024A 190 .LONG 0 ; global request ID of 0
024E 191 OP_MSG:
52 50 4F 44 4E 53 24 20 50 54 45 55 024E 192 .ASCII /UETP $SNDOPR system service test user message./
76 72 65 73 20 6D 65 74 73 79 73 20 025A
65 73 75 20 74 73 65 74 20 65 63 69 0266
2E 65 67 61 73 73 65 6D 20 72 0272
0000002E 027C 193 OP_MSG_LEN=-OP_MSG
00000036 027C 194 MSG1L=-OP_MSG1-8 ; message buffer size
027C 195 FILE_NAME:
4D 4F 43 2E 35 30 53 00' 027C 196 .ASCIC /S05.COM/
07 027C
00000008 0284 197 NAME_SIZE=-FILE_NAME
00000290 0284 198 .BLKB <FILNAMSIZ-NAME_SIZE> ; filler for SNDSMB
0290 199 FILE_NAME1:
47 4F 4C 2E 35 30 53 0290 200 .ASCII /S05.LOG/ ; log file name
0297 201 COM_FILE:
20 35 30 53 53 53 54 41 53 20 21 24 0297 202 .ASCII /$! SATSSS05 SNDSMB test batch job/<CR><LF>
20 74 73 65 74 20 42 4D 53 44 4E 53 02A3
0A 0D 62 6F 6A 20 68 63 74 61 62 02AF
21 24 02BA 203 .ASCII /$!/
00000025 02BC 204 RECO_SIZE=-COM_FILE ; record 0 size
02BC 205 REC1:
27 31 50 27 3D 3A 4D 59 53 20 24 02BC 206 .ASCII /$ SYM:='P1'/
0000000B 02C7 207 REC1_SIZE=-REC1 ; record 1 size
02C7 208 REC2:
53 51 45 2E 4D 59 53 20 46 49 20 24 02C7 209 .ASCII \ $ IF SYM.EQS.'YES' THEN DEF/GR SYM 'P1'\<CR><LF>
20 4E 45 48 54 20 22 53 45 59 22 2E 02D3
27 20 4D 59 53 20 52 47 2F 46 45 44 02DF
0A 0D 27 31 50 02EB
00000029 02F0 210 REC2_SIZE=-REC2
02F0 211 OL1:
07' 02F0 212 .BYTE OL1S
21 02F1 213 .BYTE SMOSK_HOLD
26 02F2 214 .BYTE SMOSK_PARAMS
53 45 59 00' 02F3 215 .ASCIC /YES/
03 02F3
00 02F7 216 .BYTE 0
00000007 02F8 217 OL1S=-OL1-1
02F8 218 OL2:
04' 02F8 219 .BYTE OL2S
22 02F9 220 .BYTE SMOSK_JOBPRI
03 02FA 221 .BYTE 3
21 02FB 222 .BYTE SMOSK_HOLD
00 02FC 223 .BYTE 0
00000004 02FD 224 OL2S=-OL2-1
02FD 225 JN1:
31 4D 55 4E 5F 42 4F 4A 00' 02FD 226 .ASCIC /JOB_NUM1/
08 02FD
00 0306 227 .BYTE 0
0307 228 JN2:
32 4D 55 4E 5F 42 4F 4A 00' 0307 229 .ASCIC /JOB_NUM2/
08 0307
00 0310 230 .BYTE 0
0311 231 JN3:
33 4D 55 4E 5F 42 4F 4A 00' 0311 232 .ASCIC /JOB_NUM3/
08 0311
00 031A 233 .BYTE 0
```



```
031B 235 :  
031B 236 : .SBTTL R/W PSECT  
00000000 237 : .PSECT RWDATA,RD,WRT,NOEXE, LONG  
0000 238 :  
0000 239 IPID: : PID for this process  
00000000 0000 240 : .LONG 0  
00000000 0004 241 CURRENT_TC: : ptr to current test case  
00000000 0004 242 : .LONG 0  
0008 243 : .ALIGN LONG  
0008 244 REG_SAVE_AREA: : register save area  
00000044 0008 245 : .BLKL 15  
0044 246 MOD_MSG_CODE: : test module message code for putmsg  
007480D9 0044 247 : .LONG UETPS_SATSMS  
0048 248 TMN_ADDR: :  
00000000' 0048 249 : .ADDRESS TEST_MOD_NAME  
00000019' 004C 250 TMD_ADDR: :  
0050 251 : .ADDRESS TEST_MOD_BEGIN  
0050 252 PRVPRT: :  
00 0050 253 : .BYTE 0 : protection return byte for SETPRT  
00000000 00000000 0051 254 PRIVMASK: : priv. mask  
0051 255 : .QUAD 0  
0059 256 CHM_CONT: : change mode continue address  
00000000 0059 257 : .LONG 0  
005D 258 RETADR: :  
00000065 005D 259 : .BLKL 2 : returned address's from SETPRT  
0065 260 STATUSM: :  
00000000 0065 261 : .LONG 0  
0069 262 QIO: $QIO 2,MBCHAN,IOS_READVBLK,...,BUF,BUF_SIZE+30 ; QIO parameter list  
0069 263 :  
009D 264 SNDA: $SNDACC ACC_DESC,MBCHAN ; SNDACC parameter list  
009D 265 :  
00A9 266 SNDE: $SNDERR TEST_ERROR ; SNDERR paramter list  
00A9 267 :  
00B1 268 SNDO: $SNDOPR OPMSG_DESC,0 ; SNDOPR parameter list  
00B1 269 :  
00BD 270 SNDS: $SND SMB MSG_DESC,0 ; SNDSMB paramter list  
00BD 271 :  
00C9 272 REG: :  
74 73 69 67 65 72 000000D1'010E0000' 00C9 273 : .ASCID \register R\  
52 20 72 65 00D7 :  
00DB 274 REGNUM: : register number  
00000000 00DB 275 : .LONG 0  
00DF 276 MSGL: : buffer desc.  
00000082 00DF 277 : .LONG 130  
000000E7' 00E3 278 : .ADDRESS BUF  
00E7 279 BUF: :  
00000169 00E7 280 : .BLKB 130  
0169 281 MESSAGEL: : message desc.  
00000000 0169 282 : .LONG 0  
000000E7' 016D 283 : .ADDRESS BUF  
0171 284 SERV_NAME: : service name pointer  
00000000 0171 285 : .LONG 0  
0175 286 MBCHAN: : mailbox channel number  
0000 0175 287 : .WORD 0  
0177 288 MODE: : current mode string pointer  
00000000 0177 289 : .LONG 0  
017B 290 MBUF: :
```

```
000001CB 017B 291 .BLKB 80 ; mailbox buffer
00000003 01CB 292 MSGVEC1: ; PUTMSG message vector
00741133 01CB 293 .LONG 3
00000001 01CF 294 .LONG UETPS_TEXT
00000000 01D3 295 .LONG 1
00000000 01D7 296 .LONG 0
000001E3 01DB 297 STATUS:
000001E3 01DB 298 .BLKL 2 ; mailbox status block
0001 01E3 299 ACC_MSG:
0052 01E5 300 .WORD ACC$K_INSMESG ; starting message code
00000001 01E7 301 .WORD MSG_SIZE ; message size
00000000 01EB 302 .LONG 1 ; final exit status
00000002 01EF 303 .LONG 0 ; PID
00000000 01F3 304 .LONG 2 ; job ID
54 53 00000000 00000000 01FB 305 .QUAD 0 ; system job termination time
54 53 45 54 53 59 53 00 01FB 306 .ASCIC /SYSTEST/ ; account name
07 01FB 307 .ASCIC /UETP $SNDACC system service test user data record/ ; user data
43 41 44 4E 53 24 20 50 54 45 55 00 0203
72 65 73 20 6D 65 74 73 79 73 20 43 020F
73 75 20 74 73 65 74 20 65 63 69 76 021B
6F 63 65 72 20 61 74 61 64 20 72 65 0227
64 72 0233
31 0203
00000052 0235 308 MSG_SIZE=-ACC_MSG
0006 0235 309 ACC_MSG1:
02 0237 310 .WORD ACC$K_DISASEL ; function code
11 0238 311 .BYTE ACC$K_BATTRM ; batch job type
03 0239 312 .BYTE ACC$K_INSMESG ; arbitrary message type
04 023A 313 .BYTE ACC$K_INTTRM ; interactive job type
01 023B 314 .BYTE ACC$K_LOGTRM ; login failure termination type
10 023C 315 .BYTE ACC$K_PRCTRM ; non-interactive process type
00 023D 316 .BYTE ACC$K_PRTJOB ; print job type
00000009 023E 317 .BYTE 0 ; terminator byte
00000052 023E 318 MSG1_SIZE=-ACC_MSG1
000001E3 0242 319 ACC_DESC:
000001E3 0242 320 .LONG MSG_SIZE ; descriptor for accounting message
000001E3 0242 321 .ADDRESS ACC_MSG
```



```
01000001 0246 323 OPTYPE:
02000002 0246 324 .LONG OPC$M_NM_CENTRL!<1a24> : opr type & ID table
03000004 024A 325 .LONG OPC$M_NM_PRINT!<2a24>
04000008 024E 326 .LONG OPC$M_NM_TAPES!<3a24>
05000010 0252 327 .LONG OPC$M_NM_DISKS!<4a24>
06001000 0256 328 .LONG OPC$M_NM_DEVICE!<5a24>
07002000 025A 329 .LONG OPC$M_NM_OPER1!<6a24>
08004000 025E 330 .LONG OPC$M_NM_OPER2!<7a24>
09008000 0262 331 .LONG OPC$M_NM_OPER3!<8a24>
0A010000 0266 332 .LONG OPC$M_NM_OPER4!<9a24>
0B020000 026A 333 .LONG OPC$M_NM_OPER5!<10a24>
0C040000 026E 334 .LONG OPC$M_NM_OPER6!<11a24>
0D080000 0272 335 .LONG OPC$M_NM_OPER7!<12a24>
0E100000 0276 336 .LONG OPC$M_NM_OPER8!<13a24>
0F200000 027A 337 .LONG OPC$M_NM_OPER9!<14a24>
10400000 027E 338 .LONG OPC$M_NM_OPER10!<15a24>
11800000 0282 339 .LONG OPC$M_NM_OPER11!<16a24>
12000001 0286 340 .LONG OPC$M_NM_OPER12!<17a24>
028A 341 .LONG OPC$M_NM_CENTRL!<18a24> : just to make an even number
028E 342 OPMSG_DESC:
00000080' 028E 343 .LONG MSG_LEN : SNDOPR msg buffer desc
00000296' 0292 344 .ADDRESS OPMSG
0296 345 OPMSG:
03 0296 346 .BYTE OPC$_RQ_RQST : function code
0000029A 0297 347 .BLKB 3 : operator type
00000000 029A 348 .LONG 0 : ID
00000316 029E 349 .BLKB 120 : message or terminal info
00000080 0316 350 MSG_LEN=-OPMSG
0316 351 MSG_DESC:
0000006C' 0316 352 .LONG MSG_LEN : SNDSMB msg buffer desc
0000031E' 031A 353 .ADDRESS MSG
031E 354 MSG:
0000 031E 355 .WORD SMRSK_INITIAL : SNDSMB msg buffer
00000330 0320 356 .BLKB 16 : queue name
0330 357 MSG1:
00000340 0330 358 .BLKB 16 : device name
00000346 0340 359 .BLKB 6 : file ID
0000034C 0346 360 .BLKB 6 : directory ID
00000360 034C 361 .BLKB 20 : filename
00000362 0360 362 .BLKB 2 : Job ID
0000036A 0362 363 .BLKB 8 : job name
0000038A 036A 364 .BLKB 32 : room for options and option data
0000006C 038A 365 MSG_LEN=-MSG
038A 366 SYM:
0000039E 038A 367 .BLKB 20
039E 368 :
039E 369 .ALIGN LONG
03A0 370 NAMBLK:
03A0 371 $NAM
0400 372 FAB:
0400 373 $FAB FAC=PUT,-
0400 374 FNA=FILE_NAME+1,-
0400 375 FNS=COM_FIL_SIZ,-
0400 376 NAM=NAMBLK,-
0400 377 RAT=CR,-
0400 378 RFM=VAR
0450 379 RAB:
```


SATSSS05
V04-000

- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 00:46:10 VAX/VMS Macro V04-00
R/W PSECT 5-SEP-1984 04:29:47 [UETPSY.SRC]SATSSS05.MAR;1

Page 9
(1)

0450	380		\$RAB	FAB=FAB,-
0450	381			MBF=1,-
0450	382			RBF=COM FILE,-
0450	383			RSZ=RECO_SIZE
0494	384	FAB1:		
0494	385		\$FAB	FAC=PUT,-
0494	386			FNA=FILE_NAME1,-
0494	387			FNS=COM_FIL_SIZ

```
00000000 389      .PSECT SATSSS05, RD, WRT, EXE, LONG
0000      390      .SBTTL SATSSS05
0000      391      ++
0000      392      FUNCTIONAL DESCRIPTION:
0000      393
0000      394      After performing some initial housekeeping, such as
0000      395      printing the module begin message and acquiring needed privileges,
0000      396      the system services are tested in each of their normal conditions.
0000      397      Detected failures are identified and an error message is printed
0000      398      on the terminal. Upon completion of the test a success or fail
0000      399      message is printed on the terminal.
0000      400
0000      401      CALLING SEQUENCE:
0000      402
0000      403      $ RUN SATSSS05 ... (DCL COMMAND)
0000      404
0000      405      INPUT PARAMETERS:
0000      406
0000      407      none
0000      408
0000      409      IMPLICIT INPUTS:
0000      410
0000      411      none
0000      412
0000      413      OUTPUT PARAMETERS:
0000      414
0000      415      none
0000      416
0000      417      IMPLICIT OUTPUTS:
0000      418
0000      419      Messages to SYS$OUTPUT are the only output from SATSSS05.
0000      420      They are of the form:
0000      421
0000      422      %UETP-S-SATSMS, TEST MODULE SATSSS05 BEGUN ... (BEGIN MSG)
0000      423      %UETP-S-SATSMS, TEST MODULE SATSSS05 SUCCESSFUL ... (END MSG)
0000      424      %UETP-E-SATSMS, TEST MODULE SATSSS05 FAILED ... (END MSG)
0000      425      %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
0000      426
0000      427      COMPLETION CODES:
0000      428
0000      429      The SATSSS05 routine terminates with a $EXIT to the
0000      430      operating system with a status code defined by UETP$_SATSMS.
0000      431
0000      432      SIDE EFFECTS:
0000      433
0000      434      none
0000      435
0000      436      --
0000      437
0000      438      TEST_START SATSSS05                ; let the test begin
```

```
0000 0000
0004'CF 00 DD 0002
0000'CF 02 DF 0006
00000000'GF 00 FB 000C
00000000'GF 00 FB 0013
0009'CF 01 7F 001A
00000000'GF 01 FB 001E
004C'CF 00ED4 30 0025
0044'CF 03 00 01 DE 0028
0AFE'CF 01 00 FO 002F
00 DD 0036
01 FB 0038
003D
439 STP0: .SBTTL SNDACC TESTS
440 :+
441 :
442 $SNDACC tests
443
444 test ACC$K_NEWFILE
445
446 This function will not be tested because of the possible interference
447 that it might cause with the ACCOUNTING.DAT file on a customer's system.
448
449 test ACC$K_INSMESG
450
451 :-
452
453 MODE TO,10$,KRNL,NOREGS ; kernal mode to access PHD
454 MOVL @CTL$GL PHD,R9 ; get process header address
455 MOVAL PHD$Q PRIVMSK(R9),W^PRIVMASK ; get priv mask address
456 MODE FROM,T0$ ; get back to user mode
457 PRIV ADD,OPER ; add the OPER priv.
458 MOVAL W^SNDACC,W^SERV_NAME ; set service name
459 MOVAL W^UM,W^MODE ; set the mode
460 PUSHL #0 ; push a dummy param
461 CALLS #1,W^REG_SAVE ; save a reg snapshot
462 $CREMBX_S CHAN=W^MBCHAN ; create a mailbox
463 BLBS -R0,20$ ; br if OK
464 SEXIT_S R0 ; exit and show why
465 20$:
466 $SNDACC_S MSGBUF=W^ACC_DESC,-
467 CHAN =W^MBCHAN ; try a ACC$K_NEWFILE
468 FAIL_CHECK $$$_NORMAL ; check for success
469 PUSHL #$$$_NORMAL
470 CALLS #1,W^REG_CHECK
471 CALLS #0,W^READ_CHECK ; check the mailbox
472
473 :-
474 test ACC$K_DISAACC
475
476 :-
477 NEXT_TEST
478
479 STP1:
480 MOVL #1,W^CURRENT_TC
481 PUSHL #0
```


Address	Instruction	Comment
00A5'CF 0175'CF	00E2 475 MOVW #ACCSK_DISAACC, W^ACC_MSG ; set the new function	
	00E3 476 MOVZWL W^MBCRAN, W^SNDACC_CHAN ; set up the channel number	
	00E4 477 \$SNDACC G W^SNDACC ; try ACCSK_DISAACC with a little _G	
	00E5 478 FAIL_CHECK \$\$\$_NORMAL ; check for success	
00000000'8F	00E6 479 PUSHL #\$\$\$_NORMAL	
0B08'CF 01	00E7 480 CALLS #1, W^REG_CHECK	
0C0E'CF 00	00E8 481 CALLS #0, W^READ_CHECK ; check the mailbox	
	00E9 482 :+ test ACCSK_ENABACC	
	00EA 483 :+ test ACCSK_ENABACC	
	00EB 484 :+ test ACCSK_ENABACC	
	00EC 485 :- test ACCSK_ENABACC	
	00ED 486 NEXT_TEST	
0004'CF 02	00EE 487 STP2:	
0000'CF 00	00EF 488 MOVL #2, W^CURRENT_TC	
0AFE'CF 01	00F0 489 PUSHL #0	
01E3'CF 03	00F1 490 CALLS #1, W^REG_SAVE	
	00F2 491 MOVW #ACCSK_ENABACC, W^ACC_MSG ; set function code	
	00F3 492 \$SNDACC_S MSGBOF=W^ACC_DESC,=	
	00F4 493 CHAN =W^MBCRAN ; try ACCSK_ENABACC with a little _S	
	00F5 494 FAIL_CHECK \$\$\$_NORMAL ; check for success	
00000000'8F	00F6 495 PUSHL #\$\$\$_NORMAL	
0B08'CF 01	00F7 496 CALLS #1, W^REG_CHECK	
0C0E'CF 00	00F8 497 CALLS #0, W^READ_CHECK ; check the mailbox	
	00F9 498 :+ test ACCSK_DISASEL with all types selected	
	00FA 499 :+ test ACCSK_DISASEL with all types selected	
	00FB 49A :+ test ACCSK_DISASEL with all types selected	
	00FC 49B :+ test ACCSK_DISASEL with all types selected	
	00FD 49C :+ test ACCSK_DISASEL with all types selected	
	00FE 49D :+ test ACCSK_DISASEL with all types selected	
	00FF 49E :- test ACCSK_DISASEL with all types selected	
	0100 49F NEXT_TEST	
0004'CF 03	0101 49A STP3:	
0000'CF 00	0102 49B MOVL #3, W^CURRENT_TC	
0AFE'CF 01	0103 49C PUSHL #0	
0242'CF 0235'CF	0104 49D CALLS #1, W^REG_SAVE	
023E'CF 09	0105 49E MOVAL W^ACC_MSG1, W^ACC_DESC+4 ; set new message address	
	0106 49F MOVL #MSG1_SIZE, W^ACC_DESC ; set new message size	
	0107 49A \$SNDACC G W^SNDACC ; try ACCSK_DISASEL	
	0108 49B FAIL_CHECK \$\$\$_NORMAL ; check for success	
00000000'8F	0109 49C PUSHL #\$\$\$_NORMAL	
0B08'CF 01	010A 49D CALLS #1, W^REG_CHECK	
0C0E'CF 00	010B 49E CALLS #0, W^READ_CHECK ; check the mailbox	
	010C 49F :+ test ACCSK_ENABSEL	
	010D 49A :+ test ACCSK_ENABSEL	
	010E 49B :+ test ACCSK_ENABSEL	
	010F 49C :+ test ACCSK_ENABSEL	
	0110 49D :+ test ACCSK_ENABSEL	
	0111 49E :+ test ACCSK_ENABSEL	
	0112 49F :- test ACCSK_ENABSEL	
	0113 49A NEXT_TEST	
0004'CF 04	0114 49B STP4:	
0000'CF 00	0115 49C MOVL #4, W^CURRENT_TC	
0AFE'CF 01	0116 49D PUSHL #0	
0235'CF 05	0117 49E CALLS #1, W^REG_SAVE	
	0118 49F MOVW #ACCSK_ENABSEL, W^ACC_MSG1 ; set new function	
	0119 49A \$SNDACC S MSGBOF=W^ACC_DESC,=	

00000000'8F	DD	017F	510	CHAN =W^MBCHAN	; try ACCSK_ENABSEL
0B08'CF 01	FB	018F	511	FAIL_CHECK SSS_NORMAL	; check for success
0C0E'CF 00	FB	018F		PUSHL #SS\$ NORMAL	
		0195		CALLS #1,W^REG_CHECK	
		019A	512	CALLS #0,W^READ_CHECK	; check the mailbox

.SBTTL SNDERR_S TESTS

\$SNDERR_S tests

NEXT_TEST

STP5:

```
0004'CF 05 DO 019F MOVL #5,W^CURRENT_TC
0AFE'CF 00 DD 01A4 PUSHL #0
0AFE'CF 01 FB 01A6 CALLS #1,W^REG_SAVE
0171'CF 0038'CF DE 01AB 521 PRIV ADD,BUGCHK ; add the BUGCHK priv.
0171'CF 0038'CF DE 01CB 522 MOVAL W^SNDERR,W^SERV_NAME ; set service name
0171'CF 0038'CF DE 01D2 523 $CREMBX,S CHAN=W^MBCHAN,-
0171'CF 0038'CF DE 01D2 524 LOGNAM=W^MBNAM,-
0171'CF 0038'CF DE 01D2 525 PRMFLG=#0 ; make a mailbox
0171'CF 0038'CF DE 01E9 526 $GETCHN,S CHAN=W^MBCHAN,-
0171'CF 0038'CF DE 01E9 527 PRIBUF=W^MSG ; get the unit number
0171'CF 0038'CF DE 01FF 528 MOVZWL W^BUF+DIB$W_UNIT,-(SP) ; push the MBX unit #
0171'CF 0038'CF DE 0204 529 CALLS #1,G^SYSSDERLMB ; declare errorlog MBX
0171'CF 0038'CF DE 020B 530 $SNDERR,S MSGBUF=W^TEST_ERROR ; try S form
0171'CF 0038'CF DE 0216 531 FAIL_CHECK $$$_NORMAL ; check for success
0171'CF 0038'CF DE 0216 531 PUSHL #$$$_NORMAL
0171'CF 0038'CF DE 021C 531 CALLS #1,W^REG_CHECK
0071'CF 0175'CF BO 0221 532 GET1: MOVW W^MBCHAN,W^QIO+QIOS_CHAN ; get the channel number
0071'CF 0175'CF BO 0221 533 $QIO G W^QIO ; do a read
0071'CF 0175'CF BO 0228 534 $WAITFR,S EFN=#2 ; wait for it to complete
0071'CF 0175'CF BO 0231 535 BISB2 #IOSM_NOW,W^QIO+QIOS_FUNC ; set the NOW modifier
0071'CF 0175'CF BO 023A 536 CMPW #EMBS$_SS,W^BUF+EMBS$_HD_ENTRY ; is this the right entry?
0071'CF 0175'CF BO 0240 537 BNEQ GET1 ; br if not
0071'CF 0175'CF BO 0245 538 MOVAL W^BUF+18,R6 ; set buffer address
0071'CF 0175'CF BO 0247 539 MOVAL W^TEST_ERROR+8,R7 ; set good data address
0071'CF 0175'CF BO 024C 540 MOVL #BUF_SIZE,R8 ; set byte count
0071'CF 0175'CF BO 0251 541 CALLS #0,W^BUF_CHECK ; check results
0071'CF 0175'CF BO 0258 542
```

\$SNDERR_G tests

NEXT_TEST

STP6:

```
0004'CF 06 DO 025D MOVL #6,W^CURRENT_TC
0AFE'CF 00 DD 0262 PUSHL #0
0AFE'CF 01 FB 0264 CALLS #1,W^REG_SAVE
0064 8F 00 00E7'CF 00 2C 0269 549 MOVCS #0,W^BUF,#0,#BUF_SIZE,W^BUF ; zero the buffer
0064 8F 00 00E7'CF 00 DD 0272 550 PUSHL #0 ; push a dummy parameter
0064 8F 00 00E7'CF 01 FB 0277 551 CALLS #1,W^REG_SAVE ; save a reg snapshot
0064 8F 00 00E7'CF 01 FB 027C 552 $SNDERR,G W^SNDE ; try G
0064 8F 00 00E7'CF 01 FB 0285 553 FAIL_CHECK $$$_NORMAL ; check for success
0064 8F 00 00E7'CF 01 FB 0285 553 PUSHL #$$$_NORMAL
0064 8F 00 00E7'CF 01 FB 028B 554 CALLS #1,W^REG_CHECK
0075'CF 00000000'8F CA 0290 554 BICL2 #IOSM_NOW,W^QIO+QIOS_FUNC ; set to wait for mailbox
0075'CF 00000000'8F CA 0299 555 GET2:
```


0075'CF	00'8F	88	0299	556
00EB'CF	27	B1	02A2	557
	E1	12	02AB	558
0E1A'CF	00	FB	02B1	559
	7E	D4	02B6	560
00000000'GF	01	FB	02B8	561
			02BD	562
			02BF	563

\$QIO G	W*QIO	:	read the mailbox
\$WAITFR	S EFN=#2	:	wait for completion
BISB2	#IDSM NOW,W*QIO+QIOS FUNC	:	set to read it until found
CMPL	#EMBSL_SS,W*BUF+EMBSL_HD_ENTRY	:	the right entry?
BNEQ	GET2	:	br if not
CALLS	#0,W*BUF_CHECK	:	check results
CLRL	-(SP)	:	set channel to 0
CALLS	#1,G*SYSSDERLMB	:	reset the error logger

```
.SBTTL SNDOPR TESTS
02C6 565
02C6 566 :+
02C6 567 :
02C6 568 :
02C6 569 :$SNDOPR tests
02C6 570 :
02C6 571 :DISABLE tests with _S
02C6 572 :
02C6 573 :-
02C6 574 NEXT_TEST

0004'CF 07 DO 02C6 STP7:
0000'CF 00 DD 02C6 MOVL #7,W^CURRENT_TC
0AFE'CF 01 FB 02C8 PUSHL #0
0171'CF 003F'CF DE 02CD CALLS #1,W^REG_SAVE
0296'CF 01 90 02D2 MOVAL W^SNDOPR,W^SERV_NAME ; set service name
0297'CF 8F DO 02D9 MOVW #OPCS_RQ_TERME,W^OPMSG ; set the function code
029A'CF 029E'CF D4 02DE CLRL W^OPMSG+OPCSB MS_ENAB ; set disable ID mask
029A'CF 029E'CF B4 02E2 MOVL #-1,W^OPMSG+OPCSB MS_MASK ; set operators to be disabled
02A0'CF 0239'CF 90 02EB CLRW W^OPMSG+OPCSW MS_OUNIT ; set unit to zero
02A1'CF 023A'CF DO 02EF MOVW W^OPNAME,W^OPMSG+OPCST MS_ONAME ; set operator name size
02A1'CF 023A'CF DO 02F6 MOVL W^OPNAME+1,W^OPMSG+OPCST MS_ONAME+1 ; set operator device name
02F6 581 $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
02FD 582 CHAN=W^MBCHAN ; try S
02FD 583 FAIL_CHECK SSS_NORMAL ; check success
030D 584
00000000'8F DD 030D PUSHL #SS$ NORMAL
0B08'CF 01 FB 0313 CALLS #1,W^REG_CHECK
0318 585 :+
0318 586 :
0318 587 :ENABLE tests with _S
0318 588 :
0318 589 :-
0318 590 NEXT_TEST

0004'CF 08 DO 0318 STP8:
0000'CF 00 DD 031D MOVL #8,W^CURRENT_TC
0AFE'CF 01 FB 031F PUSHL #0
0297'CF 00FFF01F 8F DO 0324 CALLS #1,W^REG_SAVE
029A'CF 00000000 8F DO 032D MOVL #ALL_OPR,W^OPMSG+OPCSB MS_ENAB ; set operators to enable
029E'CF 0154'CF B0 0336 MOVL #-1,W^OPMSG+OPCSB MS_MASK ; set enableable bits
02A0'CF 014F'CF 90 033D MOVW W^TTUNIT,W^OPMSG+OPCSW MS_OUNIT ; set the terminal unit number
02A1'CF 0150'CF DO 0344 MOVW W^TTNAM,W^OPMSG+OPCST MS_ONAME ; set the terminal name size
0344 595 MOVL W^TTNAM+1,W^OPMSG+OPCST MS_ONAME+1 ; set the terminal name
0348 596 $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
0348 597 CHAN=W^MBCHAN ; enable the alternate terminal
0358 598 FAIL_CHECK SSS_NORMAL ; check for success
0358 DD 0358 PUSHL #SS$ NORMAL
0B08'CF 01 FB 0361 CALLS #1,W^REG_CHECK
0366 599 :+
0366 600 :
0366 601 :RQST tests to make a request with ID = 1-18
0366 602 :
0366 603 :-
0366 604 NEXT_TEST

0004'CF 09 DO 0366 STP9:
0366 MOVL #9,W^CURRENT_TC
```

```
00B9' 0AFE'CF 00 DD 036B
      CF 01 FB 036D
      0175'CF 3C 0372 605
      0296'CF 03 90 0379 606
      024E'CF 2E 28 037E 607
      029E'CF 57 D4 0383 608
      029A'CF 57 D4 0386 609
      0246'CF DE 0388 610
      52 0246'CF DE 038C 611
      0297'CF 82 DO 0391 612 10$:
      CF 00 DD 0391 613
      0AFE'CF 01 FB 0396 614
      0398 615
      039D 616
      039D 617
      03AD 618
      00000000'8F DD 03AD
      0B08'CF 01 FB 03B3
      0297'CF 82 DO 03B8 619
      CF 00 DD 03BD 620
      0AFE'CF 01 FB 03BF 621
      03C4 622
      03CD 623
      00000000'8F DD 03CD
      0B08'CF 01 FB 03D3
      B5 57 09 F2 03DB 624
      03DC 625 :+
      03DC 626 :
      03DC 627 : CANCEL tests to cancel requests 1-18
      03DC 628 :
      03DC 629 :-
      03DC 630
      03DC
      0004'CF 0A DO 03DC
      CF 00 DD 03E1
      0AFE'CF 01 FB 03E3
      0296'CF 05 90 03E8 631
      52 0246'CF DE 03ED 632
      56 00008084 8F DO 03F2 633
      CF 57 D4 03F9 634
      0297'CF 82 DO 03FB 635 10$:
      CF 00 DD 03FB 636
      0AFE'CF 01 FB 0400 637
      0402 638
      0407 639
      0407 640
      0417 641
      00000000'8F DD 0417
      0B08'CF 01 FB 041D
      0E59'CF 00 FB 0422 642
      0297'CF 82 DO 0427 643
      CF 00 DD 042C 644
      0AFE'CF 01 FB 042E 645
      0433 646
      043C 647
      00000000'8F DD 043C

      PUSHL #0
      CALLS #1,W^REG_SAVE
      MOVZWL W^MBCHAN,W^SNDO+SNDOPRS_CHAN : set the channel number
      MOVW #OPCS_RQ_RQST,W^OPMSG : set function code
      MOVW #OP_MESG_LEN,W^OP_MESG,-
      W^OPMSG+OPCSL_MS_TEXT : put the text in the message
      CLRL R7 : init loop variable
      CLRL W^OPMSG+OPCSL_MS_RQSTID : init the ID field
      MOVAL W^OPTYE,R2 : set opr type list pointer

      MOVW (R2)+,W^OPMSG+1 : set opr type & ID
      PUSHL #0 : push a dummy parameter
      CALLS #1,W^REG_SAVE : save the registers
      $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
      CHAN=W^MBCHAN : try S form
      FAIL_CHECK $$$_NORMAL : check for success
      PUSHL #$$$_NORMAL
      CALLS #1,W^REG_CHECK
      MOVW (R2)+,W^OPMSG+1 : set opr type & ID
      PUSHL #0 : push a dummy param
      CALLS #1,W^REG_SAVE : save a reg snapshot
      $SNDOPR G W^SNDO : try G
      FAIL_CHECK $$$_NORMAL : check for success
      PUSHL #$$$_NORMAL
      CALLS #1,W^REG_CHECK
      AOBLSS #9,R7,10$ : do all opr types

      :+
      :
      : CANCEL tests to cancel requests 1-18
      :
      :-
      NEXT_TEST

      STP10:
      MOVW #10,W^CURRENT_TC
      PUSHL #0
      CALLS #1,W^REG_SAVE
      MOVW #OPCS_RQ_CANCEL,W^OPMSG : set function code
      MOVAL W^OPTYE,R2 : set table pointer
      MOVW #<OPCS_RQSTCAN&^XFFFF>,R6 : set completion code
      CLRL R7 : set loop variable

      MOVW (R2)+,W^OPMSG+1 : set opr type & ID
      PUSHL #0 : push a dummy parameter
      CALLS #1,W^REG_SAVE : save a reg snapshot
      $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
      CHAN=W^MBCHAN : try S form
      FAIL_CHECK $$$_NORMAL : check success
      PUSHL #$$$_NORMAL
      CALLS #1,W^REG_CHECK
      CALLS #0,W^SND_CHECK : check the results
      MOVW (R2)+,W^OPMSG+1 : set opr type & ID
      PUSHL #0 : push a dummy parameter
      CALLS #1,W^REG_SAVE : save a reg snapshot
      $SNDOPR G W^SNDO : try G form
      FAIL_CHECK $$$_NORMAL : check success
      PUSHL #$$$_NORMAL
```


	OB08'CF	01	FB	0442		CALLS #1,W^REG_CHECK	
	OE59'CF	00	FB	0447	648	#0,W^SND CHECK	; check the results
	AB 57	09	F2	044C	649	AOBLSS #9,R7,108	; do all opr types
				0450	650	:	
				0450	651	:	
				0450	652	:+ REPLY tests to respond to requests	
				0450	653	:	
				0450	654	:-	
				0450	655	NEXT_TEST	
				0450			
				0450			
	0004'CF	0B	D0	0450		STP11:	
		00	DD	0455		MOVL #11,W^CURRENT_TC	
	OAFE'CF	01	FB	0457		PUSHL #0	
56	00008029	8F	D0	045C	656	CALLS #1,W^REG_SAVE	
	0296'CF	04	90	0463	657	MOVW #<OPCS RQSTCMPLTE&*XFFFF>,R6	; set expected status return
	0298'CF	56	B0	0468	658	MOVW R6,W^OPMSG+OPCSW_MS_STATUS	; set the function
	029A'CF		D4	046D	659	CLRL W^OPMSG+OPCSL_MS_RPYID	; set status reply return
029E'CF	0154'CF		B0	0471	660	MOVW W^TTUNIT,W^OPMSG+OPCSW_MS_OUNIT	; set the message ID
02AO'CF	014F'CF	05	28	0478	661	MOVW W^TTNAM,W^OPMSG+OPCSL_MS_ONAME	; set the unit number
	024E'CF	2E	28	0480	662	MOVW #5,W^TTNAM,W^OPMSG+OPCSL_MS_ONAME	; set the device name
	02B0'CF			0485	663	MOVW #OP MSG LEN,W^OP MSG,-	
		00	DD	0488	664	W^OPMSG+OPCSL_MS_OTEXT	; set the message text
	OAFE'CF	01	FB	048A	665	PUSHL #0	; push a dummy parameter
	OEB1'CF	00	FB	048F	666	CALLS #1,W^REG_SAVE	; save a reg snapshot
				0494	667	CALLS #0,W^GENREQ	; generate a pending request
				0494	668	\$SNDOPR_S MSGBUF=W^OPMSG_DESC,-	
				04A4	669	CHAN=W^MBCHAN	; try S
	00000000'	8F	DD	04A4		FAIL_CHECK \$\$\$ NORMAL	; check success
	OB08'CF	01	FB	04AA		PUSHL #\$\$\$ NORMAL	
	OE59'CF	00	FB	04AF	670	CALLS #1,W^REG_CHECK	
56	801C	8F	B0	04B4	671	CALLS #0,W^SND CHECK	; check results
	0298'CF	56	B0	04B9	672	MOVW #<OPCS RQSTABORT&*XFFFF>,R6	; set expected status return
		00	DD	04BE	673	MOVW R6,W^OPMSG+OPCSW_MS_STATUS	; set reply status code
	OAFE'CF	01	FB	04C0	674	PUSHL #0	; push a dummy parameter
	OEB1'CF	00	FB	04C5	675	CALLS #1,W^REG_SAVE	; save a reg snapshot
				04CA	676	CALLS #0,W^GENREQ	; generate a pending request
				04D3	677	\$SNDOPR_G W^SNDO	; try G
	00000000'	8F	DD	04D3		FAIL_CHECK \$\$\$ NORMAL	; check success
	OB08'CF	01	FB	04D9		PUSHL #\$\$\$ NORMAL	
	OE59'CF	00	FB	04DE	678	CALLS #1,W^REG_CHECK	
56	8021	8F	B0	04E3	679	CALLS #0,W^SND CHECK	; check results
	0298'CF	56	B0	04E8	680	MOVW #<OPCS RQSTPEND&*XFFFF>,R6	; set expected status return
		00	DD	04ED	681	MOVW R6,W^OPMSG+OPCSW_MS_STATUS	; set the reply status code
	OAFE'CF	01	FB	04EF	682	PUSHL #0	; push a dummy parameter
	OEB1'CF	00	FB	04F4	683	CALLS #1,W^REG_SAVE	; save a reg snapshot
				04F9	684	CALLS #0,W^GENREQ	; generate a pending request
				04F9	685	\$SNDOPR_S MSGBUF=W^OPMSG_DESC,-	
				0509	686	CHAN=W^MBCHAN	; try S and leave the request pendi
	00000000'	8F	DD	0509		FAIL_CHECK \$\$\$ NORMAL	; check success
	OB08'CF	01	FB	050F		PUSHL #\$\$\$ NORMAL	
	OE59'CF	00	FB	0514	687	CALLS #1,W^REG_CHECK	
56	8084	8F	B0	0519	688	CALLS #0,W^SND CHECK	; check results
	0298'CF	56	B0	051E	689	MOVW #<OPCS RQSTCAN&*XFFFF>,R6	; set expected status return
		00	DD	0523	690	MOVW R6,W^OPMSG+OPCSW_MS_STATUS	; set reply status
	OAFE'CF	01	FB	0523	691	PUSHL #0	; push a dummy parameter
				052A	692	CALLS #1,W^REG_SAVE	; save a reg snapshot
						\$SNDOPR_G W^SNDO	; try G

```

00000000'8F DD 0533 693 FAIL_CHECK $$$_NORMAL ; check success
0B08'CF 01 FB 0533 PUSHL #$$$ NORMAL
0E59'CF 00 FB 0539 CALLS #1,W^REG_CHECK
                                ; check results
                                :+
                                :+
                                :+
0543 694 CALLS #0,W^SND_CHECK
0543 695 :+
0543 696 :+
0543 697 :+ DISABLE tests with _G
0543 698 :+
0543 699 :-
0543 700 NEXT_TEST
0543
STP12:
0004'CF 0C DO 0543 MOVL #12,W^CURRENT_TC
0000'CF 00 DD 0548 PUSHL #0
0AFE'CF 01 FB 054A CALLS #1,W^REG_SAVE
0296'CF 01 90 054F 701 MOVBL #OPCS_RQ_TERME,W^OPMSG ; set the function code
0297'CF D4 0554 702 CLRL W^OPMSG+OPCSB_MS_ENAB ; set disable ID mask
029A'CF FFFFFFFF 8F DO 0558 703 MOVL #-1,W^OPMSG+OPCSL_MS_MASK ; set operators to disable
                                ; disable the alternate TTY
                                ; check for success
00000000'8F DD 0561 704 $$SNDOPR G W^SND
0B08'CF 01 FB 056A 705 FAIL_CHECK $$$_NORMAL
                                PUSHL #$$$ NORMAL
                                CALLS #1,W^REG_CHECK
                                :+
                                :+
                                :+
0575 706 :+
0575 707 :+ ENABLE tests with _G
0575 708 :+
0575 709 :+
0575 710 :-
0575 711 NEXT_TEST
0575
STP13:
0004'CF 0D DO 0575 MOVL #13,W^CURRENT_TC
0000'CF 00 DD 057A PUSHL #0
0AFE'CF 01 FB 057C CALLS #1,W^REG_SAVE
0297'CF 00FFF01F 8F DO 0581 712 MOVBL #ALL_OPR,W^OPMSG+OPCSB_MS_ENAB ; set enable ID mask
029A'CF FFFFFFFF 8F DO 058A 713 MOVL #-1,W^OPMSG+OPCSL_MS_MASK ; set all enables
                                ; set unit number
02A0'CF 029E'CF B4 0593 714 CLRW W^OPMSG+OPCSW_MS_UNIT
02A1'CF 0239'CF 90 0597 715 MOVBL W^OPNAME,W^OPMSG+OPCST_MS_ONAME ; set name size
02A1'CF 023A'CF DO 059E 716 MOVL W^OPNAME+1,W^OPMSG+OPCST_MS_ONAME+1 ; set name
                                ; enable the console again
                                ; check for failure
00000000'8F DD 05A5 717 $$SNDOPR G W^SND
0B08'CF 01 FB 05AE 718 FAIL_CHECK $$$_NORMAL
                                PUSHL #$$$ NORMAL
                                CALLS #1,W^REG_CHECK

```

```
05B9 720 .SBTTL SNDSMB TESTS
05B9 721 :+
05B9 722 :
05B9 723 $SNDSMB tests
05B9 724 :
05B9 725 The following request types cannot be tested because of the lack of a
05B9 726 queueable device in the minimum configuration.
05B9 727 :
05B9 728 SMR&K_ABORT, SMR&K_ASSIGN, SMR&K_JUSTIFY, SMR&K_ENTER
05B9 729 :
05B9 730 test SMR&K_INITIAL by creating que1 and que2
05B9 731 :
05B9 732 :-
05B9 733 NEXT_TEST
05B9
05B9 STP14:
05B9 MOVL #14,W^CURRENT_TC
05BE PUSHL #0
05C0 CALLS #1,W^REG_SAVE
05C5 MOVAL W^SNDSMB,W^SERV_NAME ; set service name
05C6 MOVW W^MBCHAN,W^SNDS+SNDSMB$_CHAN ; set the mailbox channel #
05C7 MOVAL W^MSG+2,R3 ; set argument pointer
05C8 MOVCL #QUENAM1L,W^QUENAM1,(R3) ; set the queue name
05C9 MOVAL W^MSG1,R3 ; set to proper end of que name
05CA MOVW #SMR&K_DETJOB,(R3)+ ; set to BATCH
05CB CLRB (R3) ; set option terminator
05CD PUSHL #0 ; push a dummy parameter
05CE CALLS #1,W^REG_SAVE ; save a reg snapshot
05CF $SNDSMB$_MSGBUF = W^MSG DESC,-
05D0 CHAN = W^MBCHAN ; try S INITIAL
05D1 FAIL_CHECK SS$NORMAL ; check failure
0600 PUSHL #SS$NORMAL
0606 CALLS #1,W^REG_CHECK
060B MOVL #JBC$NORMAL,R6 ; set expected return status
060C CALLS #0,W^SND_CHECK ; check results
060D MOVW #SMR&K_DSWAP,(R3)+ ; set to disable swapping
060E MOVW #SMR&K_INIPRI,(R3)+ ; set a new job priority
060F MOVW #2,(R3)+ ; by default of 2
0610 MOVW #SMR&K_JOBLIM,(R3)+ ; set a job limit of
0611 MOVW #2,(R3)+ ; 2
0612 CLRB (R3) ; set the terminator
0613 MOVCL #QUENAM2L,W^QUENAM2,W^MSG+2 ; set new que name
0614 PUSHL #0 ; push a dummy parameter
0615 CALLS #1,W^REG_SAVE ; save a register snapshot
0616 $SNDSMB G W^SNDS ; init the next que
0617 FAIL_CHECK SS$NORMAL ; check for failure
0618 PUSHL #SS$NORMAL
0619 CALLS #1,W^REG_CHECK
061A CALLS #0,W^SND_CHECK ; check the results
061B :+
061C :
061D :
061E :
061F :
0620 :
0621 :
0622 :
0623 :
0624 :
0625 :
0626 :
0627 :
0628 :
0629 :
062A :
062B :
062C :
062D :
062E :
062F :
0630 :
0631 :
0632 :
0633 :
0634 :
0635 :
0636 :
0637 :
0638 :
0639 :
063A :
063B :
063C :
063D :
063E :
063F :
0640 :
0641 :
0642 :
0643 :
0644 :
0645 :
0646 :
0647 :
0648 :
0649 :
064A :
064B :
064C :
064D :
064E :
064F :
0650 :
0651 :
0652 :
0653 :
0654 :
0655 :
0656 :
0657 :
0658 :
0659 :
065A :
065B :
065C :
065D :
065E :
065F :
0660 :
0661 :
0662 :
0663 :
0664 :
0665 :
0666 :
0667 :
0668 :
0669 :
066A :
066B :
066C :
066D :
066E :
066F :
0670 :
0671 :
0672 :
0673 :
0674 :
0675 :
0676 :
0677 :
0678 :
0679 :
067A :
067B :
067C :
067D :
067E :
067F :
0680 :
0681 :
0682 :
0683 :
0684 :
0685 :
0686 :
0687 :
0688 :
0689 :
068A :
068B :
068C :
068D :
068E :
068F :
0690 :
0691 :
0692 :
0693 :
0694 :
0695 :
0696 :
0697 :
0698 :
0699 :
069A :
069B :
069C :
069D :
069E :
069F :
06A0 :
06A1 :
06A2 :
06A3 :
06A4 :
06A5 :
06A6 :
06A7 :
06A8 :
06A9 :
06AA :
06AB :
06AC :
06AD :
06AE :
06AF :
06B0 :
06B1 :
06B2 :
06B3 :
06B4 :
06B5 :
06B6 :
06B7 :
06B8 :
06B9 :
06BA :
06BB :
06BC :
06BD :
06BE :
06BF :
06C0 :
06C1 :
06C2 :
06C3 :
06C4 :
06C5 :
06C6 :
06C7 :
06C8 :
06C9 :
06CA :
06CB :
06CC :
06CD :
06CE :
06CF :
06D0 :
06D1 :
06D2 :
06D3 :
06D4 :
06D5 :
06D6 :
06D7 :
06D8 :
06D9 :
06DA :
06DB :
06DC :
06DD :
06DE :
06DF :
06E0 :
06E1 :
06E2 :
06E3 :
06E4 :
06E5 :
06E6 :
06E7 :
06E8 :
06E9 :
06EA :
06EB :
06EC :
06ED :
06EE :
06EF :
06F0 :
06F1 :
06F2 :
06F3 :
06F4 :
06F5 :
06F6 :
06F7 :
06F8 :
06F9 :
06FA :
06FB :
06FC :
06FD :
06FE :
06FF :
0700 :
0701 :
0702 :
0703 :
0704 :
0705 :
0706 :
0707 :
0708 :
0709 :
070A :
070B :
070C :
070D :
070E :
070F :
0710 :
0711 :
0712 :
0713 :
0714 :
0715 :
0716 :
0717 :
0718 :
0719 :
071A :
071B :
071C :
071D :
071E :
071F :
0720 :
0721 :
0722 :
0723 :
0724 :
0725 :
0726 :
0727 :
0728 :
0729 :
072A :
072B :
072C :
072D :
072E :
072F :
0730 :
0731 :
0732 :
0733 :
0734 :
0735 :
0736 :
0737 :
0738 :
0739 :
073A :
073B :
073C :
073D :
073E :
073F :
0740 :
0741 :
0742 :
0743 :
0744 :
0745 :
0746 :
0747 :
0748 :
0749 :
074A :
074B :
074C :
074D :
074E :
074F :
0750 :
0751 :
0752 :
0753 :
0754 :
0755 :
0756 :
0757 :
0758 :
0759 :
075A :
075B :
075C :
075D :
075E :
075F :
0760 :
0761 :
0762 :
0763 :
0764 :
0765 :
0766 :
0767 :
0768 :
0769 :
076A :
076B :
076C :
076D :
076E :
076F :
0770 :
0771 :
0772 :
0773 :
0774 :
0775 :
0776 :
0777 :
0778 :
0779 :
077A :
077B :
077C :
077D :
077E :
077F :
0780 :
0781 :
0782 :
0783 :
0784 :
0785 :
0786 :
0787 :
0788 :
0789 :
078A :
078B :
078C :
078D :
078E :
078F :
0790 :
0791 :
0792 :
0793 :
0794 :
0795 :
0796 :
0797 :
0798 :
0799 :
079A :
079B :
079C :
079D :
079E :
079F :
07A0 :
07A1 :
07A2 :
07A3 :
07A4 :
07A5 :
07A6 :
07A7 :
07A8 :
07A9 :
07AA :
07AB :
07AC :
07AD :
07AE :
07AF :
07B0 :
07B1 :
07B2 :
07B3 :
07B4 :
07B5 :
07B6 :
07B7 :
07B8 :
07B9 :
07BA :
07BB :
07BC :
07BD :
07BE :
07BF :
07C0 :
07C1 :
07C2 :
07C3 :
07C4 :
07C5 :
07C6 :
07C7 :
07C8 :
07C9 :
07CA :
07CB :
07CC :
07CD :
07CE :
07CF :
07D0 :
07D1 :
07D2 :
07D3 :
07D4 :
07D5 :
07D6 :
07D7 :
07D8 :
07D9 :
07DA :
07DB :
07DC :
07DD :
07DE :
07DF :
07E0 :
07E1 :
07E2 :
07E3 :
07E4 :
07E5 :
07E6 :
07E7 :
07E8 :
07E9 :
07EA :
07EB :
07EC :
07ED :
07EE :
07EF :
07F0 :
07F1 :
07F2 :
07F3 :
07F4 :
07F5 :
07F6 :
07F7 :
07F8 :
07F9 :
07FA :
07FB :
07FC :
07FD :
07FE :
07FF :
0800 :
0801 :
0802 :
0803 :
0804 :
0805 :
0806 :
0807 :
0808 :
0809 :
080A :
080B :
080C :
080D :
080E :
080F :
0810 :
0811 :
0812 :
0813 :
0814 :
0815 :
0816 :
0817 :
0818 :
0819 :
081A :
081B :
081C :
081D :
081E :
081F :
0820 :
0821 :
0822 :
0823 :
0824 :
0825 :
0826 :
0827 :
0828 :
0829 :
082A :
082B :
082C :
082D :
082E :
082F :
0830 :
0831 :
0832 :
0833 :
0834 :
0835 :
0836 :
0837 :
0838 :
0839 :
083A :
083B :
083C :
083D :
083E :
083F :
0840 :
0841 :
0842 :
0843 :
0844 :
0845 :
0846 :
0847 :
0848 :
0849 :
084A :
084B :
084C :
084D :
084E :
084F :
0850 :
0851 :
0852 :
0853 :
0854 :
0855 :
0856 :
0857 :
0858 :
0859 :
085A :
085B :
085C :
085D :
085E :
085F :
0860 :
0861 :
0862 :
0863 :
0864 :
0865 :
0866 :
0867 :
0868 :
0869 :
086A :
086B :
086C :
086D :
086E :
086F :
0870 :
0871 :
0872 :
0873 :
0874 :
0875 :
0876 :
0877 :
0878 :
0879 :
087A :
087B :
087C :
087D :
087E :
087F :
0880 :
0881 :
0882 :
0883 :
0884 :
0885 :
0886 :
0887 :
0888 :
0889 :
088A :
088B :
088C :
088D :
088E :
088F :
0890 :
0891 :
0892 :
0893 :
0894 :
0895 :
0896 :
0897 :
0898 :
0899 :
089A :
089B :
089C :
089D :
089E :
089F :
08A0 :
08A1 :
08A2 :
08A3 :
08A4 :
08A5 :
08A6 :
08A7 :
08A8 :
08A9 :
08AA :
08AB :
08AC :
08AD :
08AE :
08AF :
08B0 :
08B1 :
08B2 :
08B3 :
08B4 :
08B5 :
08B6 :
08B7 :
08B8 :
08B9 :
08BA :
08BB :
08BC :
08BD :
08BE :
08BF :
08C0 :
08C1 :
08C2 :
08C3 :
08C4 :
08C5 :
08C6 :
08C7 :
08C8 :
08C9 :
08CA :
08CB :
08CC :
08CD :
08CE :
08CF :
08D0 :
08D1 :
08D2 :
08D3 :
08D4 :
08D5 :
08D6 :
08D7 :
08D8 :
08D9 :
08DA :
08DB :
08DC :
08DD :
08DE :
08DF :
08E0 :
08E1 :
08E2 :
08E3 :
08E4 :
08E5 :
08E6 :
08E7 :
08E8 :
08E9 :
08EA :
08EB :
08EC :
08ED :
08EE :
08EF :
08F0 :
08F1 :
08F2 :
08F3 :
08F4 :
08F5 :
08F6 :
08F7 :
08F8 :
08F9 :
08FA :
08FB :
08FC :
08FD :
08FE :
08FF :
0900 :
0901 :
0902 :
0903 :
0904 :
0905 :
0906 :
0907 :
0908 :
0909 :
090A :
090B :
090C :
090D :
090E :
090F :
0910 :
0911 :
0912 :
0913 :
0914 :
0915 :
0916 :
0917 :
0918 :
0919 :
091A :
091B :
091C :
091D :
091E :
091F :
0920 :
0921 :
0922 :
0923 :
0924 :
0925 :
0926 :
0927 :
0928 :
0929 :
092A :
092B :
092C :
092D :
092E :
092F :
0930 :
0931 :
0932 :
0933 :
0934 :
0935 :
0936 :
0937 :
0938 :
0939 :
093A :
093B :
093C :
093D :
093E :
093F :
0940 :
0941 :
0942 :
0943 :
0944 :
0945 :
0946 :
0947 :
0948 :
0949 :
094A :
094B :
094C :
094D :
094E :
094F :
0950 :
0951 :
0952 :
0953 :
0954 :
0955 :
0956 :
0957 :
0958 :
0959 :
095A :
095B :
095C :
095D :
095E :
095F :
0960 :
0961 :
0962 :
0963 :
0964 :
0965 :
0966 :
0967 :
0968 :
0969 :
096A :
096B :
096C :
096D :
096E :
096F :
0970 :
0971 :
0972 :
0973 :
0974 :
0975 :
0976 :
0977 :
0978 :
0979 :
097A :
097B :
097C :
097D :
097E :
097F :
0980 :
0981 :
0982 :
0983 :
0984 :
0985 :
0986 :
0987 :
0988 :
0989 :
098A :
098B :
098C :
098D :
098E :
098F :
0990 :
0991 :
0992 :
0993 :
0994 :
0995 :
0996 :
0997 :
0998 :
0999 :
0A00 :
0A01 :
0A02 :
0A03 :
0A04 :
0A05 :
0A06 :
0A07 :
0A08 :
0A09 :
0A0A :
0A0B :
0A0C :
0A0D :
0A0E :
0A0F :
0A10 :
0A11 :
0A12 :
0A13 :
0A14 :
0A15 :
0A16 :
0A17 :
0A18 :
0A19 :
0A1A :
0A1B :
0A1C :
0A1D :
0A1E :
0A1F :
0A20 :
0A21 :
0A22 :
0A23 :
0A24 :
0A25 :
0A26 :
0A27 :
0A28 :
0A29 :
0A2A :
0A2B :
0A2C :
0A2D :
0A2E :
0A2F :
0A30 :
0A31 :
0A32 :
0A33 :
0A34 :
0A35 :
0A36 :
0A37 :
0A38 :
0A39 :
0A3A :
0A3B :
0A3C :
0A3D :
0A3E :
0A3F :
0A40 :
0A41 :
0A42 :
0A43 :
0A44 :
0A45 :
0A46 :
0A47 :
0A48 :
0A49 :
0A4A :
0A4B :
0A4C :
0A4D :
0A4E :
0A4F :
0A50 :
0A51 :
0A52 :
0A53 :
0A54 :
0A55 :
0A56 :
0A57 :
0A58 :
0A59 :
0A5A :
0A5B :
0A5C :
0A5D :
0A5E :
0A5F :
0A60 :
0A61 :
0A62 :
0A63 :
0A64 :
0A65 :
0A66 :
0A67 :
0A68 :
0A69 :
0A6A :
0A6B :
0A6C :
0A6D :
0A6E :
0A6F :
0A70 :
0A71 :
0A72 :
0A73 :
0A74 :
0A75 :
0A76 :
0A77 :
0A78 :
0A79 :
0A7A :
0A7B :
0A7C :
0A7D :
0A7E :
0A7F :
0A80 :
0A81 :
0A82 :
0A83 :
0A84 :
0A85 :
0A86 :
0A87 :
0A88 :
0A89 :
0A8A :
0A8B :
0A8C :
0A8D :
0A8E :
0A8F :
0A90 :
0A91 :
0A92 :
0A93 :
0A94 :
0A95 :
0A96 :
0A97 :
0A98 :
0A99 :
0AA0 :
0AA1 :
0AA2 :
0AA3 :
0AA4 :
0AA5 :
0AA6 :
0AA7 :
0AA8 :
0AA9 :
0AAB :
0AAC :
0AAD :
0AAE :
0AAF :
0AB0 :
0AB1 :
0AB2 :
0AB3 :
0AB4 :
0AB5 :
0AB6 :
0AB7 :
0AB8 :
0AB9 :
0ABA :
0ABB :
0ABC :
0ABD :
0ABE :
0ABF :
0AC0 :
0AC1 :
0AC2 :
0AC3 :
0AC4 :
0AC5 :
0AC6 :
0AC7 :
0AC8 :
0AC9 :
0ACA :
0ACB :
0ACC :
0ACD :
0ACE :
0ACF :
0AD0 :
0AD1 :
0AD2 :
0AD3 :
0AD4 :
0AD5 :
0AD6 :
0AD7 :
0AD8 :
0AD9 :
0ADA :
0ADB :
0ADC :
0ADD :
0ADE :
0ADF :
0AE0 :
0AE1 :
0AE2 :
0AE3 :
0AE4 :
0AE5 :
0AE6 :
0AE7 :
0AE8 :
0AE9 :
0AEA :
0AEB :
0AEC :
0AED :
0AEE :
0AEF :
0AF0 :
0AF1 :
0AF2 :
0AF3 :
0AF4 :
0AF5 :
0AF6 :
0AF7 :
0AF8 :
0AF9 :
0AFA :
0AFB :
0AFC :
0AFD :
0AFE :
0AFF :
0B00 :
0B01 :
0B02 :
0B03 :
0B04 :
0B05 :
0B06 :
0B07 :
0B08 :
0B09 :
0B0A :
0B0B :
0B0C :
0B0D :
0B0E :
0B0F :
0B10 :
0B11 :
0B12 :
0B13 :
0B14 :
0B15 :
0B16 :
0B17 :
0B18 :
0B19 :
0B1A :
0B1B :
0B1C :
0B1D :
0B1E :
0B1F :
0B20 :
0B21 :
0B22 :
0B23 :
0B24 :
0B25 :
0B26 :
0B27 :
0B28 :
0B29 :
0B2A :
0B2B :
0B2C :
0B2D :
0B2E :
0B2F :
0B30 :
0B31 :
0B32 :
0B33 :
0B34 :
0B35 :
0B36 :
0B37 :
0B38 :
0B39 :
0B3A :
0B3B :
0B3C :
0B3D :
0B3E :
0B3F :
0B40 :
0B41 :
0B42 :
0B43 :
0B44 :
0B45 :
0B46 :
0B47 :
0B48 :
0B49 :
0B4A :
0B4B :
0B4C :
0B4D :
0B4E :
0B4F :
0B50 :
0B51 :
0B52 :
0B53 :
0B54 :
0B55 :
0B56 :
0B57 :
0B58 :
0B59 :
0B5A :
0B5B :
0B5C :
0B5D :
0B5E :
0B5F :
0B60 :
0B61 :
0B62 :
0B63 :
0B64 :
0B65 :
0B66 :
0B67 :
0B68 :
0B69 :
0B6A :
0B6B :
0B6C :
0B6D :
0B6E :
0B6F :
0B70 :
0B71 :
0B72 :
0B73 :
0B74 :
0B75 :
0B76 :
0B77 :
0B78 :
0B79 :
0B7A :
0B7B :
0B7C :
0B7D :
0B7E :
0B7F :
0B80 :
0B81 :
0B82 :
0B83 :
0B84 :
0B85 :
0B86 :
0B87 :
0B88 :
0B89 :
0B8A :
0B8B :
0B8C :
0B8D :
0B8E :
0B8F :
0B90 :
0B91 :
0B92 :
0B93 :
0B94 :
0B95 :
0B96 :
0B97 :
0B98 :
0B99 :
0BA0 :
0BA1 :
0BA2 :
0BA3 :
0BA4 :
0BA5 :
0BA6 :
0BA7 :
0BA8 :
0BA9 :
0BAA :
0BAB :
0BAC :
0BAD :
0BAE :
0BAF :
0BB0 :
0BB1 :
0BB2 :
0BB3 :
0BB4 :
0BB5 :
0BB6 :
0BB7 :
0BB8 :
0BB9 :
0BBA :
0BBB :
0BBC :
0BBD :
0BBE :
0BBF :
0BC0 :
0BC1 :
0BC2 :
0BC3 :
0BC4 :
0BC5 :
0BC6 :
0BC7 :
0BC8 :
0BC9 :
0BCA :
0BCB :
0BCC :
0BCD :
0BCE :
0BCF :
0BD0 :
0BD1 :
0BD2 :
0BD3 :
0BD4 :
0BD5 :
0BD6 :
0BD7 :
0BD8 :
0BD9 :
0BDA :
0BDB :
0BDC :
0BDD :
0BDE :
0BDF :
0BE0 :
0BE1 :
0BE2 :
0BE3 :
0BE4 :
0BE5 :
0BE6 :
0BE7 :
0BE8 :
0BE9 :
0BEA :
0BEB :
0BEC :
0BED :
0BEE :
0BEF :
0BF0 :
0BF1 :
0BF2 :
0BF3 :
0BF4 :
0BF5 :
0BF6 :
0BF7 :
0BF8 :
0BF9 :
0BFA :
0BFB :
0BFC :
0BFD :
0BFE :
0BFF :
0C00 :
0C01 :
0C02 :
0C03 :
0C04 :
0C05 :
0C06 :
0C07 :
0C08 :
0C09 :
0C0A :
0C0B :
0C0C :
0C0D :
0C0E :
0C0F :
0C10 :
0C11 :
0C12 :
0C13 :
0C14 :
0C15 :
0C16 :
0C17 :
0C18 :
0C19 :
0C1A :
0C1B :
0C1C :
0C1D :
0C1E :
0C1F :
0C20 :
0C21 :
0C22 :
0C23 :
0C24 :
0C25 :
0C26 :
0C27 :
0C28 :
0C29 :
0C2A :
0C2B :
0C2C :
0C2D :
0C2E :
0C2F :
0C30 :
0C31 :
0C32 :
0C33 :
0C34 :
0C35 :
0C36 :
0C37 :
0C38 :
0C39 :
0C3A :
0C3B :
0C3C :
0C3D :
0C3E :
0C3F :
0C40 :
0C41 :
0C42 :
0C43 :
0C44 :
0C45 :
0C46 :
0C47 :
0C48 :
0C49 :
0C4A :
0C4B :
0C4C :
0C4D :
0C4E :
0C4F :
0C50 :
0C51 :
0C52 :
0C53 :
0C54 :
0C55 :
0C56 :
0C57 :
0C58 :
0C59 :
0C5A :
0C5B :
0C5C :
0C5D :
0C5E :
0C5F :
0C60 :
0C61 :
0C62 :
0C63 :
0C64 :
0C65 :
0C66 :
0C67 :
0C68 :
0C69 :
0C6A :
0C6B :
0C6C :
0C6D :
0C6E :
0C6F :
0C70 :
0C71 :
0C72 :
0C73 :
0C74 :
0C75 :
0C76 :
0C77 :
0C78 :
0C79 :
0C7A :
0C7B :
0C7C :
0C7D :
0C7E :
0C7F :
0C80 :
0C81 :
0C82 :
0C83 :
0C84 :
0C85 :
0C86 :
0C87 :
0C88 :
0C89 :
0C8A :
0C8B :
0C8C :
0C8D :
0C8E :
0C8F :
0C90 :
0C91 :
0C92 :
0C93 :
0C94 :
0C95 :
0C96 :
0C97 :
0C98 :
0C99 :
0CA0 :
0CA1 :
0CA2 :
0CA3 :
0CA4 :
0CA5 :
0CA6 :
0CA7 :
0CA8 :
0CA9 :
0CAA :
0CAB :
0CAC :
0CAD :
0CAE :
0CAF :
0CB0 :
0CB1 :
0CB2 :
0CB3 :
0CB4 :
0CB5 :
0CB6 :
0CB7 :
0CB8 :
0CB9 :
0CBA :
0CBB :
0CBC :
0CBD :
0CBE :
0CBF :
0CC0 :
0CC1 :
0CC2 :
0CC3 :
0CC4 :
0CC5 :
0CC6 :
0CC7 :
0CC8 :
0CC9 :
0CCA :
0CCB :
0CCC :
0CCD :
0CCE :
0CCF :
0CD0 :
0CD1 :
0CD2 :
0CD3 :
0CD4 :
0CD5 :
0CD6 :
0CD7 :
0CD8 :
0CD9 :
0CDA :
0CDB :
0CDC :
0CDD :
0CDE :
0CDF :
0CE0 :
0CE1 :
0CE2 :
0CE3 :
0CE4 :
0CE5 :
0CE6 :
0CE7 :
0CE8 :
0CE9 :
0CEA :
0CEB :
0CEC :
0CED :
0CEE :
0CEF :
0CF0 :
0CF1 :
0CF2 :
0CF3 :
0CF4 :
0CF5 :
0CF6 :
0CF7 :
0CF8 :
0CF9 :
0CFA :
0CFB :
0CFC :
0CFD :
0CFE :
0CFF :
0D00 :
0D01 :
0D02 :
0D03 :
0D04 :
0D05 :
0D06 :
0D07 :
0D08 :
0D09 :
0D0A :
0D0B :
0D0C :
0D0D :
0D0E :
0D0F :
0D10 :
0D11 :
0D12 :
0D13 :
0D14 :
0D15 :
0D16 :
0D17 :
0D18 :
0D19 :
0D1A :
0D1B :
0D1C :
0D1D :
0D1E :
0D1F :
0D20 :
0D21 :
0D22 :
0D23 :
0D24 :
0D25 :
0D26 :
0D27 :
0D28 :
0D29 :
0D2A :
0D2B :
0D2C :
0D2D :
0D2E :
0D2F :
0D30 :
0D31 :
0D32 :
0D33 :
0D34 :
0D35 :
0D36 :
0D37 :
0D38 :
0D39 :
0D3A :
0D3B :
0D3C :
0D3D :
0D3E :
0D3F :
0D40 :
0D41 :
0D42 :
0D43 :
0D44 :
0D45 :
0D46 :
0D47 :
0D48 :
0D49 :
0D4A :
0D4B :
0D4C :
0D4D :
0D4E :
0D4F :
0D50 :
0D51 :
0D52 :
0D53 :
0D54 :
0D55 :
0D56 :
0D57 :
0D58 :
0D59 :
0D5A :
0D5B :
0D5C :
0D5D :
0D5E :
0D5F :
0D60 :
0D61 :
0D62 :
0D63 :
0D64 :
0D65 :
0D66 :
0D67 :
0D68 :
0D69 :
0D6A :
0D6B :
0D6C :
0D6D :
0D6E :
0D6F :
0D70 :
0D71 :
0D72 :
0D73 :
0D74 :
0D75 :
0D76 :
0D77 :
0D78 :
0D79 :
0D7A :
0D7B :
0D7C :
0D7D :
0D7E :
0D7F :
0D80 :
0D81 :
0D82 :
0D83 :
0D84 :
0D85 :
0D86 :
0D87 :
0D88 :
0D89 :
0D8A :
0D8B :
0D8C :
0D8D :
0D8E :
0D8F :
0D90 :
0D91 :
0D92 :
0D93 :
0D94 :
0D95 :
0D96 :
0D97 :
0D98 :
0D99 :
0DA0 :
0DA1 :
0DA2 :
0DA3 :
0DA4 :
0DA5 :
0DA6 :
0DA7 :
0DA8 :
0DA9 :
0DAA :
0DAB :
0DAC :
0DAD :
0DAE :
0DAF :
0DB0 :
0DB1 :
0DB2 :
0DB3 :
0DB4 :
0DB5 :
0DB6 :
0DB7 :
0DB8 :
0DB9 :
0DBA :
0DBB :
0DBC :
0DBD :
0DBE :
0DBF :
0DC0 :
0DC1 :
0DC2 :
0DC3 :
0DC4 :
0DC5 :
0DC6 :
0DC7 :
0DC8 :
0DC9 :
0DCA :
0DCB :
0DCC :
0DCD :
0DCE :
0DCF :
0DD0 :
0DD1 :
0DD2 :
0DD3 :
0DD4 :
0DD5 :
0DD6 :
0DD7 :
0DD8 :
0DD9 :
0DDA :
0DDB :
0DDC :
0DDD :
0DDE :
0DDF :
0DE0 :
0DE1 :
0DE2 :
0DE3 :
0DE4 :
0DE5 :
0DE6 :
0DE7 :
0DE8 :
0DE9 :
0DEA :
0DEB :
0DEC :
0DED :
0DEE :
0DEF :
0DF0 :
0DF1 :
0DF2 :
0DF3 :
0DF4 :
0DF5 :
0DF6 :
0DF7 :
0DF8 :
0DF9 :
0DFA :
0DFB :
0DFC :
0DFD :
0DFE :
0DFF :
0E00 :
0E01 :
0E02 :
0E03 :
0E04 :
0E05 :
0E06 :
0E07 :
0E08 :
0E09 :
0E0A :
0E0B :
0E0C :
0E0D :
0E0E :
0E0F :
0E10 :
0E11 :
0E12 :
0E13 :
0E14 :
0E15 :
0E16 :
0E17 :
0E18 :
0E19 :
0E1A :
0E1B :
0E1C :
0E1D :
0E1E :
0E1F :
0E20 :
0E21 :
0E22 :
0E23 :
0E24 :
0E25 :
0E26 :
0E27 :
0E28 :
0E29 :
0E2A :
0E2B :
0E2C :
0E2D :
0E2E :
0E2F :
0E30 :
0E31 :
0E32 :
0E33 :
0E34 :
0E35 :
0E36 :
0E37 :
0E38 :
0E39 :
0E3A :
0E3B :
0E3C :
0E3D :
0E3E :
0E3F :
0E40 :
0E41 :
0E42 :
0E43 :
0E44 :
0E45 :
0E46 :
0E47 :
0E48 :
0E49 :
0E4A :
0E4B :
0E4C :
0E4D :
0E4E :
0E4F :
0E50
```



```
0004'CF 0F DO 0653      MOVL #15,W^CURRENT_TC
0000'CF 00 DD 0658      PUSHL #0
0AFE'CF 01 FB 065A      CALLS #1,W^REG_SAVE
031E'CF 02 BO 065F 766  MOVW #SMR$K_START,W^MSG      ; set request code
0330'CF 94 0664 767  CLRB W^MSGT      ; set for no start options
00000000'8F DD 0668 768  $SNDSMB G W^SNDS      ; try G START
0B08'CF 01 FB 0671 769  FAIL_CHECK SSS_NORMAL      ; check failure
0E59'CF 00 FB 0677      PUSHL #SS$ NORMAL
01A1'CF 0E 28 067C 770  CALLS #1,W^REG_CHECK      ; check results
0AFE'CF 01 FB 0677      MOVW #0,W^SND_CHECK      ; set que name
00000000'8F DD 0681 771  CALLS #1,W^REG_SAVE      ; push a dummy param
0B08'CF 01 FB 0689 772  $SNDSMB G W^SNDS      ; save a reg snapshot
0E59'CF 00 FB 068B 773  FAIL_CHECK SSS_NORMAL      ; start the next que
00000000'8F DD 0690 774  CALLS #1,W^REG_CHECK      ; check for failures
0B08'CF 01 FB 0699 775  PUSHL #SS$ NORMAL
0E59'CF 00 FB 06A4 776  CALLS #0,W^SND_CHECK      ; check the results
00000000'8F DD 06A9 777  ;+
0B08'CF 01 FB 06A9 778  ; test SMR$K_STOP
0E59'CF 00 FB 06A9 779  ;
00000000'8F DD 06A9 780  ;
0B08'CF 01 FB 06A9 781  ;
0E59'CF 00 FB 06A9 782  ; -
0004'CF 10 DO 06A9      STP16:
0000'CF 00 DD 06AE      MOVL #16,W^CURRENT_TC
0AFE'CF 01 FB 06B0      PUSHL #0
031E'CF 07 BO 06B5 783  CALLS #1,W^REG_SAVE      ; set request code
00000000'8F DD 06B8 784  MOVW #SMR$K_STOP,W^MSG      ; try S STOP
0B08'CF 01 FB 06BA 785  $SNDSMB S MSGBOF = W^MSG DESC,-      ; check failure
0E59'CF 02 BO 06BA 786  CHAN = W^MBCHAN
00000000'8F DD 06CA      FAIL_CHECK SSS_NORMAL
0B08'CF 01 FB 06CA      PUSHL #SS$ NORMAL
0E59'CF 00 FB 06D0      CALLS #1,W^REG_CHECK      ; check results
00000000'8F DD 06D5 787  CALLS #0,W^SND_CHECK      ; reset request code
0B08'CF 01 FB 06DA 788  MOVW #SMR$K_START,W^MSG      ; restart the queue
0E59'CF 02 BO 06DF 789  $SNDSMB G W^SNDS      ; check failure
00000000'8F DD 06E8 790  FAIL_CHECK SSS_NORMAL
0B08'CF 01 FB 06E8      PUSHL #SS$ NORMAL
0E59'CF 00 FB 06EE      CALLS #1,W^REG_CHECK      ; check results
00000000'8F DD 06F3 791  CALLS #0,W^SND_CHECK
0B08'CF 01 FB 06F8 792  ;+
0E59'CF 00 FB 06F8 793  ; test SMR$K_PAUSE
00000000'8F DD 06F8 794  ;
0B08'CF 01 FB 06F8 795  ;
0E59'CF 00 FB 06F8 796  ; -
0004'CF 11 DO 06F8      STP17:
0000'CF 00 DD 06FD      MOVL #17,W^CURRENT_TC
0AFE'CF 01 FB 06FF      PUSHL #0
031E'CF 03 BO 0704 798  CALLS #1,W^REG_SAVE      ; set request code
00000000'8F DD 0709 799  MOVW #SMR$K_PAUSE,W^MSG      ; try S PAUSE
0B08'CF 01 FB 0709 800  $SNDSMB S MSGBOF = W^MSG DESC,-      ; check failure
0E59'CF 00 FB 0719 801  CHAN = W^MBCHAN
00000000'8F DD 0719      FAIL_CHECK SSS_NORMAL
```

	00000000'8F	DD	0719	PUSHL #SS\$ NORMAL	
	0B08'CF 01	FB	071F	CALLS #1,W^REG_CHECK	
	0E59'CF 00	FB	0724	CALLS #0,W^SND_CHECK	; check results
	031E'CF 02	BO	0729	MOVW #SMR\$K START,W^MSG	; reset the request code
			072E	\$SNDSMB G W^SNS	; reset the queue state
			0737	FAIL_CHECK SS\$ NORMAL	; check failure
	00000000'8F	DD	0737	PUSHL #SS\$ NORMAL	
	0B08'CF 01	FB	073D	CALLS #1,W^REG_CHECK	
	0E59'CF 00	FB	0742	CALLS #0,W^SND_CHECK	; check results
			0747		
			0747		
			0747	test SMR\$K_CREJOB, SMR\$K_ADDFIL, SMR\$K_CLSJOB	
			0747		
			0747		
			0747	NEXT_TEST	
			0747		
			0747	STP18:	
	0004'CF 12	DO	0747	MOVL #18,W^CURRENT_TC	
		DD	074C	PUSHL #0	
	0AFE'CF 01	FB	074E	CALLS #1,W^REG_SAVE	
			0753	\$CREATE FAB = W^FAB	; open the command file
			075E	\$CONNECT RAB = W^RAB	; connect up
			0769	\$PUT RAB = W^RAB	; write the command file
0478'CF 02BC'CF	DE	0774	816	MOVAL W^REC1,W^RAB+RAB\$L RBF	; set rec #1 address
0472'CF 0B	BO	077B	817	MOVW #REC1 SIZE,W^RAB+RAB\$W_RSZ	; set rec #1 size
			0780	\$PUT RAB = W^RAB	; write record #1
0478'CF 02C7'CF	DE	078B	819	MOVAL W^REC2,W^RAB+RAB\$L RBF	; set rec #2 address
0472'CF 29	BO	0792	820	MOVW #REC2 SIZE,W^RAB+RAB\$W_RSZ	; set rec #2 size
			0797	\$PUT RAB = W^RAB	; write record #2
			07A2	\$DISCONNECT RAB = W^RAB	; disconnect
			07AD	\$CLOSE FAB = W^FAB	; file S05.COM now exists
	02F0'CF	DF	07B8	PUSHAL W^OL1	; set option list #1
	02FD'CF	DF	07BC	PUSHAL W^JN1	; set job name #1
OCFD'CF 02	FB	07C0	826	CALLS #2,W^CRE_JOB	; put a job in the que
			07C5		
			07C5		
			07C5	test SMR\$K_ALTER on job #1 to release it	
			07C5		
			07C5		
			07C5	NEXT_TEST	
			07C5		
			07C5	STP19:	
	0004'CF 13	DO	07C5	MOVL #19,W^CURRENT_TC	
		DD	07CA	PUSHL #0	
	0AFE'CF 01	FB	07CC	CALLS #1,W^REG_SAVE	
031E'CF 0D	BO	07D1	833	MOVW #SMR\$K ALTER,W^MSG	; set request code
53 0320'CF	DE	07D6	834	MOVAL W^MSGT2,R3	; set message buffer pointer
01A1'CF 0E	28	07DB	835	MOVC3 #QUENAM1L,W^QUENAM1,(R3)	; set the que name
53 0330'CF	DE	07E1	836	MOVAL W^MSGI,R3	; set to correct end of que name
83 OCE7'CF	BO	07E6	837	MOVW W^JOBID,(R3)+	; set job ID
83 22	90	07EB	838	MOVB #SMO\$K_JOBPRI,(R3)+	; set option code
83 01	90	07EE	839	MOVB #1,(R3)+	; set the job priority
	63	94	07F1	CLRB (R3)	; terminate the option list
	00	DD	07F3	PUSHL #0	; push a dummy parameter
0AFE'CF 01	FB	07F5	842	CALLS #1,W^REG_SAVE	; save a register snapshot
			07FA	\$SNDSMB G W^SNS	; try G ALTER
			0803	FAIL_CHECK SS\$ NORMAL	; check failure

```
00000000'8F DD 0803          PUSHL  #SS$ NORMAL
0B08'CF 01 FB 0809          CALLS  #1,W*REG_CHECK
0E59'CF 00 FB 080E          CALLS  #0,W*SNDS_CHECK          ; check results
                        845
                        846 :+
                        847 : test SMR$K_RELEASE on Job #1
                        848 :
                        849 :-
                        850
                        851 NEXT_TEST
                        851
                        851 STP20:
0004'CF 14 DO 0813          MOVL  #20,W*CURRENT_TC
0AFE'CF 01 DD 0818          PUSHL  #0
031E'CF 0F FB 081A          CALLS  #1,W*REG_SAVE
0330'CF 0CE7'CF BO 081F 852 MOVW  #SMR$K_RELEASE,W*MSG          ; set request code
0332'CF 94 BO 0824 853 MOVW  W*JOBID,W*MSG1          ; set job ID
                        854 CLRB  W*MSG1+2          ; set no options
                        855 $SNDSMB G W*SNDS          ; try G RELEASE
                        856 FAIL_CHECK SSS$ NORMAL          ; check failure
00000000'8F DD 0838          PUSHL  #SS$ NORMAL
0B08'CF 01 FB 083E          CALLS  #1,W*REG_CHECK
0E59'CF 00 FB 0843          CALLS  #0,W*SNDS_CHECK          ; check results
                        857
                        858 :+
                        859 : test SMR$K_SYNCJOB on Job #1
                        860 :
                        861 :-
                        862
                        863 NEXT_TEST
                        863
                        863 STP21:
0004'CF 15 DO 0848          MOVL  #21,W*CURRENT_TC
0AFE'CF 01 DD 084D          PUSHL  #0
031E'CF 11 FB 084F          CALLS  #1,W*REG_SAVE
0330'CF 0CE7'CF BO 0854 864 MOVW  #SMR$K_SYNCJOB,W*MSG          ; set request code
0332'CF 94 BO 0859 865 MOVW  W*JOBID,W*MSG1          ; set job ID
                        866 CLRB  W*MSG1+2          ; set option list end
                        867 $SNDSMB G W*SNDS          ; try G SYNCJOB
                        868 FAIL_CHECK SSS$ NORMAL          ; check failure
00000000'8F DD 086D          PUSHL  #SS$ NORMAL
0B08'CF 01 FB 0873          CALLS  #1,W*REG_CHECK
56 00000000'8F DO 0878 869 MOVL  #SS$ NORMAL,R6          ; set expected status return
56 0E59'CF 00 FB 087F 870 CALLS  #0,W*SNDS_CHECK          ; check results
00040001 8F DO 0884 871 MOVL  #JBC$ NORMAL,R6          ; set expected status return
                        872 $TRNLOG_S LOGNAM = W*YES_DESC,-
                        873 RSLBUF = W*SYM_DESC,-
                        874 DSBMSK = #5
                        875 CMPL  #SS$ NORMAL,R0          ; look for the group symbol
                        876 BEQL  10$,          ; is it there?
                        877 PUSHAL W*BAT_IMP_EXC          ; br if OK
                        878 CALLS  #1,W*PRINT_FAIL          ; push error message address
                        879 10$:          ; print the failure
                        880 $DELLOG S LOGNAM = W*YES_DESC          ; dump the logical name
                        881 $ERASE FAB = W*FAB1          ; delete the log file
                        882 :+
                        883 :
                        884 : test SMR$K_RMVJOB on Job #2
                        885 :
```



```
08CE 886 :-  
08CE 887 NEXT_TEST  
08CE  
08CE STP22:  
0004'CF 16 DO 08CE  
00 DD 08D3  
0AFE'CF 01 FB 08D5  
02F8'CF DF 08DA 888  
0307'CF DF 08DE 889  
OCFD'CF 02 FB 08E2 890  
031E'CF 0C BO 08E7 891  
0330'CF 0CE7'CF BO 08EC 892  
0332'CF 94 08F3 893  
00 DD 08F7 894  
0AFE'CF 01 FB 08F9 895  
08FE 896  
0907 897  
00000000'8F DD 0907  
0B08'CF 01 FB 090D  
0E59'CF 00 FB 0912 898  
0917 899 :+  
0917 900 : test SMR$K_MERGE on job #3  
0917 901 :  
0917 902 :  
0917 903 :-  
0917 904 NEXT_TEST  
0917 STP23:  
0004'CF 17 DO 0917  
00 DD 091C  
0AFE'CF 01 FB 091E  
02F0'CF DF 0923 905  
0311'CF DF 0927 906  
OCFD'CF 02 FB 092B 907  
53 031E'CF DE 0930 908  
83 04 BO 0935 909  
63 01AF'CF 0E 28 0938 910  
53 0330'CF DE 093E 911  
63 01A1'CF 0E 28 0943 912  
02 A3 94 0949 913  
00 DD 094C 914  
0AFE'CF 01 FB 094E 915  
0953 916  
095C 917  
00000000'8F DD 095C  
0B08'CF 01 FB 0962  
0E59'CF 00 FB 0967 918  
53 031E'CF DE 096C 919  
83 0F BO 0971 920  
63 01AF'CF 0E 28 0974 921  
53 0330'CF DE 097A 922  
83 0CE7'CF BO 097F 923  
63 94 0984 924  
00 DD 0986 925  
0AFE'CF 01 FB 0988 926  
098D 927  
0996 928  
MOV#L #23,W^CURRENT_TC  
PUSH#L #0  
CALLS #1,W^REG_SAVE  
PUSH#L W^OL2  
PUSH#L W^JN2  
CALLS #2,W^CRE_JOB  
MOV#W #SMR$K_RM#JOB,W^MSG  
MOV#W W^JOBID,W^MSG1  
CLRB W^MSG1+2  
PUSH#L #0  
CALLS #1,W^REG_SAVE  
$SNDSMB G W^SNDS  
FAIL_CHECK $$$_NORMAL  
PUSH#L #$$$_NORMAL  
CALLS #1,W^REG_CHECK  
CALLS #0,W^SND_CHECK  
: set option list #2  
: set job name #2  
: put job #2 in the que  
: set request code  
: set job ID  
: set no options  
: push a dummy parameter  
: save a reg snapshot  
: try G and nail the last job  
: check failure  
: check results  
MOV#L #23,W^CURRENT_TC  
PUSH#L #0  
CALLS #1,W^REG_SAVE  
PUSH#L W^OL1  
PUSH#L W^JN3  
CALLS #2,W^CRE_JOB  
MOV#L W^MSG,R3  
MOV#W #SMR$K_MERGE,(R3)+  
MOV#C3 #QUENAM2L,W^QUENAM2,(R3)  
MOV#L W^MSG1,R3  
MOV#C3 #QUENAM1L,W^QUENAM1,(R3)  
CLRB 2(R3)  
PUSH#L #0  
CALLS #1,W^REG_SAVE  
$SNDSMB G W^SNDS  
FAIL_CHECK $$$_NORMAL  
PUSH#L #$$$_NORMAL  
CALLS #1,W^REG_CHECK  
CALLS #0,W^SND_CHECK  
MOV#L W^MSG,R3  
MOV#W #SMR$K_RELEASE,(R3)+  
MOV#C3 #QUENAM2L,W^QUENAM2,(R3)  
MOV#L W^MSG1,R3  
MOV#W W^JOBID,(R3)+  
CLRB (R3)  
PUSH#L #0  
CALLS #1,W^REG_SAVE  
$SNDSMB G W^SNDS  
FAIL_CHECK $$$_NORMAL  
: set option list #3  
: set job name #3  
: put job 3 in the que  
: set address  
: set request code  
: set queue name 1  
: get to correct end of name  
: set queue name 2  
: set no options(*watch que name len  
: push a dummy parameter  
: save a reg snapshot  
: try G MERGE  
: check failure  
: check results  
: set message address  
: set request code  
: set the que name  
: get to the end of the quenam  
: set the job ID  
: set no options  
: push a dummy parameter  
: save a register snapshot  
: release the job  
: check for failures
```

```
00000000'8F DD 0996          PUSHL  #SS$ NORMAL
0B08'CF 01 FB 099C          CALLS  #1,W^REG_CHECK
0E59'CF 00 FB 09A1 929      CALLS  #0,W^SND_CHECK          ; check the results
031E'CF 11 BO 09A6 930      MOVW  #SMR$K SYNCJOB,W^MSG      ; set request code
                                $SNDSMB G W^SNDS          ; sync on the job
                                FAIL_CHECK SSS$ NORMAL    ; check for failures
00000000'8F DD 09B4          PUSHL  #SS$ NORMAL
0B08'CF 01 FB 09BA          CALLS  #1,W^REG_CHECK
56 00000000'8F DO 09BF 933      MOVL  #SS$ NORMAL,R6          ; set the expected status return
0E59'CF 00 FB 09C6 934      CALLS  #0,W^SND_CHECK          ; check the results
56 00040001 8F DO 09CB 935      MOVL  #JBC$ NORMAL,R6          ; set the expected status return
                                $STRNLOG_S LOGNAM = W^YES_DESC,-
                                RSLBUF = W^SYM_DESC,-
                                DSBMSK = #5
00000000'8F D1 09E9 939      CMPL  #SS$ NORMAL,R0          ; look for the group symbol
09 13 09F0 940      BEQL  10$,
0164'CF DF 09F2 941      PUSHAL W^BAT IMP_EXC          ; is it there?
0B4A'CF 01 FB 09F6 942      CALLS  #1,W^PRINT_FAIL        ; br if OK
                                10$:
                                $DELLOG_S LOGNAM = W^YES_DESC ; push error message address
                                ; print the failure
                                09FB 943
                                09FB 944
                                0A0A 945 ;+
                                0A0A 946 ; test SMR$K_DELETE
                                0A0A 947
                                0A0A 948
                                0A0A 949 ;-
                                0A0A 950
                                NEXT_TEST
STP24:
0004'CF 18 DO 0A0A          MOVL  #24,W^CURRENT_TC
00 DD 0A0F          PUSHL  #0
0AFE'CF 01 FB 0A11          CALLS  #1,W^REG_SAVE
0320'CF 031E'CF 07 BO 0A16 951      MOVW  #SMR$K STOP,W^MSG      ; set request code
01A1'CF 0E 28 0A1B 952      MOVW  #QUENAM1L,W^QUENAM1,W^MSG+2 ; set the que name
0330'CF 94 0A23 953      CLRB  W^MSG1          ; set no options
00 DD 0A27 954      PUSHL  #0          ; push a dummy parameter
0AFE'CF 01 FB 0A29 955      CALLS  #1,W^REG_SAVE          ; save a reg snapshot
                                $SNDSMB G W^SNDS          ; stop que 1
                                0A2E 956
                                0A37 957      FAIL_CHECK SSS$ NORMAL ; check for failures
                                PUSHL  #SS$ NORMAL
                                CALLS  #1,W^REG_CHECK
00000000'8F DD 0A37          CALLS  #0,W^SND_CHECK          ; check the results
0B08'CF 01 FB 0A3D          MOVW  #QUENAM2L,W^QUENAM2,W^MSG+2 ; set the que name
0320'CF 0E59'CF 00 FB 0A42 958      CALLS  #0,W^SND_CHECK          ; check the results
01AF'CF 0E 28 0A47 959      MOVW  #QUENAM2L,W^QUENAM2,W^MSG+2 ; set the que name
00 DD 0A4F 960      PUSHL  #0          ; push a dummy param
0AFE'CF 01 FB 0A51 961      CALLS  #1,W^REG_SAVE          ; save a reg snapshot
                                $SNDSMB G W^SNDS          ; stop que 2
                                0A56 962
                                0A5F 963      FAIL_CHECK SSS$ NORMAL ; check for failures
                                PUSHL  #SS$ NORMAL
                                CALLS  #1,W^REG_CHECK
00000000'8F DD 0A5F          CALLS  #0,W^SND_CHECK          ; check the results
0B08'CF 01 FB 0A65          MOVW  #SMR$K DELETE,W^MSG      ; set the request code
0E59'CF 00 FB 0A6A 964      $SNDSMB G W^SNDS          ; delete the que
031E'CF 01 BO 0A6F 965      FAIL_CHECK SSS$ NORMAL          ; check for failures
                                0A74 966
                                0A7D 967      PUSHL  #SS$ NORMAL
                                CALLS  #1,W^REG_CHECK
00000000'8F DD 0A7D          CALLS  #0,W^SND_CHECK          ; check the results
0B08'CF 01 FB 0A83          MOVW  #QUENAM1L,W^QUENAM1,W^MSG+2 ; set the new que name
0320'CF 0E59'CF 00 FB 0A88 968      CALLS  #0,W^SND_CHECK          ; check the results
01A1'CF 0E 28 0A8D 969      MOVW  #QUENAM1L,W^QUENAM1,W^MSG+2 ; set the new que name
00 DD 0A95 970      PUSHL  #0          ; push a dummy parameter
```

Page 26
(2)

Address	Operation	Comments
0AFE'CF 01 FB	QA97 971	CALLS #1,W^REG_SAVE ; save a reg snapshot
	QA9C 972	\$SNDSMB G W^SNDS ; delete the last que
	QAA5 973	FAIL_CHECK \$\$\$_NORMAL ; check for failures
00000000'8F DD	QAA5	PUSHL #\$\$\$ NORMAL
0B08'CF 01 FB	QAAB	CALLS #1,W^REG_CHECK
0E59'CF 00 FB	QAB0 974	CALLS #0,W^SND_CHECK ; check the last results
	QAB5 975	\$ERASE FAB=W^FAB ; delete the .COM file
	QAC0 976	\$ERASE FAB=W^FAB1 ; delete the .LOG file
	QACB 977	\$DASSGN S CHAN=W^MBCHAN ; drop the mailbox
	QAD7 978	TEST_END
004C'CF DD	QAD7	PUSHL W^TMD_ADDR
0048'CF DD	QADB	PUSHL W^TMN_ADDR
02 DD	QADF	PUSHL #2
0044'CF DD	QAE1	PUSHL W^MOD_MSG_CODE
00000000'GF 04 FB	QAE5	CALLS \$\$\$T1,G^LTB\$\$SIGNAL
0044'CF 01 1C 01 FO	QAE5	INSV #1,\$\$SSV_INHIB_MSG,#1,W^MOD_MSG_CODE
0044'CF DD	QAF3	PUSHL W^MOD_MSG_CODE
00000000'GF 01 FB	QAF7	CALLS #1,G^SYSSEXIT

[illegible]


```
0008'CF 14 AD 28 OFFC 04
0008'CF 14 AD 28 29
56 53 00000008'8F C3
7E 56 04 C6
51 03 CA
53 03 CA

0AFE 980 .SBTTL REG_SAVE
0AFE 981 :++
0AFE 982 FUNCTIONAL DESCRIPTION:
0AFE 983 Subroutine to save R2-R11 in the register save location.
0AFE 984
0AFE 985 CALLING SEQUENCE:
0AFE 986 PUSHL #0 ; save a dummy parameter
0AFE 987 CALLS #1,W^REG_SAVE ; save R2-R11
0AFE 988
0AFE 989 INPUT PARAMETERS:
0AFE 990 NONE
0AFE 991
0AFE 992 OUTPUT PARAMETERS:
0AFE 993 NONE
0AFE 994
0AFE 995 :--
0AFE 996
0AFE 997 REG_SAVE:
0AFE 998 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0B00 999 MOVC3 #4*10,^X14(FP),W^REG_SAVE_AREA ; save the registers in the program
0B07 1000 RET
0B08 1001 .SBTTL REG_CHECK
0B08 1002 :++
0B08 1003 FUNCTIONAL DESCRIPTION:
0B08 1004 Subroutine to test R0 & R2-R11 for proper content after a service
0B08 1005 execution. A snapshot is taken by the REG_SAVE routine at the
0B08 1006 beginning of each step and this routine is executed after the
0B08 1007 services have been executed.
0B08 1008
0B08 1009 CALLING SEQUENCE:
0B08 1010 PUSHL #SS$ XXXXXX ; push expected R0 contents
0B08 1011 CALLS #1,W^REG_CHECK ; execute this routine
0B08 1012
0B08 1013 INPUT PARAMETERS:
0B08 1014 expected R0 contents on the stack
0B08 1015
0B08 1016 OUTPUT PARAMETERS:
0B08 1017 possible error messages printed using $PUTMSG
0B08 1018
0B08 1019 :--
0B08 1020
0B08 1021 REG_CHECK:
0B08 1022 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0B0A 1023 CMPL 4(AP),R0 ; is this the right fail code?
0B0E 1024 BEQL 10$ ; br if yes
0B10 1025 PUSHL R0 ; push received data
0B12 1026 PUSHL 4(AP) ; push expected data
0B15 1027 PUSHAL W^EXP ; push the string variable
0B19 1028 CALLS #3,W^PRINT_FAIL ; print the error message
0B1E 1029 10$:
0B1E 1030 CMPC3 #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0
0B25 1031 BEQL 20$ ; br if O.K.
0B27 1032 SUBL3 #REG_SAVE_AREA,R3,R6 ; calculate the register number
0B2F 1033 DIVL2 #4,R6
0B32 1034 ADDB3 #^X2,R6,-(SP) ; set number past R0-R1 and save
0B36 1035 BICL2 #3,R1 ; backup to register boundrys
0B39 1036 BICL2 #3,R3
```

```
61 DD OB3C 1037          PUSHL (R1)          ; push received data
63 DD OB3E 1038          PUSHL (R3)          ; push expected data
00C9'CF DF OB40 1039          PUSHAL W^REG      ; set string pntr param.
OB4A'CF 04 FB OB44 1040          CALLS #4,W^PRINT_FAIL ; print the error message
                                20$:
                                04 OB49 1041          RET
                                OB4A 1042          .SBTTL PRINT_FAIL
                                OB4A 1043          :++
                                OB4A 1044          : FUNCTIONAL DESCRIPTION:
                                OB4A 1045          : Subroutine to report failures using $PUTMSG
                                OB4A 1046          :
                                OB4A 1047          : CALLING SEQUENCE:
                                OB4A 1048          : Mode #1          PUSHL EXPECTED Mode #2          PUSHL REG NUMBER
                                OB4A 1049          : Mode #1          PUSHL RECEIVED          PUSHL EXPECTED
                                OB4A 1050          :                   PUSHAL STRING VAR          PUSHL RECEIVED
                                OB4A 1051          :                   CALLS #3,W^PRINT_FAIL          PUSHAL STRING VAR
                                OB4A 1052          :                   CALLS #4,W^PRINT_FAIL          CALLS #4,W^PRINT_FAIL
                                OB4A 1053          :
                                OB4A 1054          : Mode #3          PUSHAL STRING VAR
                                OB4A 1055          :                   CALLS #1,W^PRINT_FAIL
                                OB4A 1056          :
                                OB4A 1057          : INPUT PARAMETERS:
                                OB4A 1058          : Listed above
                                OB4A 1059          :
                                OB4A 1060          : OUTPUT PARAMETERS:
                                OB4A 1061          : an error message is printed using $PUTMSG
                                OB4A 1062          :
                                OB4A 1063          :--
                                OB4A 1064          :
                                003C OB4A 1065          PRINT_FAIL:
                                OB4A 1066          .WORD ^M<R2,R3,R4,R5>
                                OB4C 1067          $FAO S W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                                OB6D 1068          $PUTMSG_S W^MSGVEC          ; print the message
                                04 6C 91 OB7E 1069          CMPB (AP),#4          ; is this a register message?
                                01 26 13 OB81 1070          BEQL 10$          ; br if yes
                                01 6C 91 OB83 1071          CMPB (AP),#1          ; is this just a message?
                                48 13 OB86 1072          BEQL 20$          ; br if yes
                                40 11 OB88 1073          $FAO_S W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                                OBA7 1074          BRB 30$          ; goto output message
                                OBA9 1075          10$:
                                OBA9 1076          $FAO_S W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                                19 11 OBCE 1077          BRB 30$          ; goto output message
                                01D7'CF 04 AC D0 OBDO 1078          20$:
                                OBDO 1079          MOVL 4(AP),W^MSGVEC1+12          ; save string address
                                OBDO 1080          $PUTMSG_S W^MSGVEC1          ; print the message
                                11 11 OBE7 1081          BRB 40$          ; skip the other message
                                OBE9 1082          30$:
                                OBE9 1083          $PUTMSG_S W^MSGVEC          ; print the message
                                OBFA 1084          40$:
                                00ECF'CF 00 FB OBFA 1085          CALLS #0,W^MODE_ID          ; identify the mode
                                004C'CF 002A'CF DE OBFF 1086          MOVAL W^TEST_MOD_FAIL,W^TMD_ADDR          ; set failure message address
                                0044'CF 03 00 02 FO OC06 1087          INSV #ERROR,#0,#3,W^MOD_MSG_CODE          ; set severity code
                                04 OC0D 1088          RET
                                OC0E 1089          .SBTTL READ_CHECK
                                OC0E 1090          :++
                                OC0E 1091          : FUNCTIONAL DESCRIPTION:
                                OC0E 1092          : Subroutine to read a mailbox and check the status returned
                                OC0E 1093          : from the $SNDACC system service.
```

```
0000'8F 017B'CF B1 0C0E 1094 : CALLING SEQUENCE:
                                0C0E 1095 : CALLS #0,W^READ_CHECK
                                0C0E 1096 :
                                0C0E 1097 : INPUT PARAMETERS:
                                0C0E 1098 : NONE
                                0C0E 1099 :
                                0C0E 1100 : OUTPUT PARAMETERS:
                                0C0E 1101 : NONE
                                0C0E 1102 :
                                0C0E 1103 :
                                0C0E 1104 : --
                                0C0E 1105 :
003C 0C0E 1106 READ_CHECK:
                                0C0E 1107 WORD ^M<R2,R3,R4,R5>
                                0C10 1108 $QIOW_S FUNC=#IOS$ READVBLK,-
                                0C10 1109 CHAN=W^MBCHAN,-
                                0C10 1110 IOSB=W^STATUSM,-
                                0C10 1111 P1 =W^MBUF,-
                                0C10 1112 P2 =#80
                                0C37 1113 CMPW W^MBUF,#MSG$_ACCRSP : read the mail
                                0C3E 1114 BEQL 10$ : correct response type?
                                0C40 1115 PUSHL W^MBUF : br if yes
                                0C44 1116 PUSHL #MSG$_ACCRSP : push received
                                0C4A 1117 PUSHAL W^EXP : push expected
                                0C4E 1118 CALLS #3,W^PRINT_FAIL : push string variable
                                0C53 1119 10$: : print the failure
                                0C53 1120 CMPL W^MBUF+4,#4216!SS$_NORMAL : check the results
                                0C5C 1121 BEQL 20$ : br if OK
                                0C5E 1122 PUSHL W^MBUF+4 : push received
                                0C62 1123 PUSHL #4216!SS$_NORMAL : push expected
                                0C68 1124 PUSHAL W^EXP : push the string variable
                                0C6C 1125 CALLS #3,W^PRINT_FAIL : print the failure
                                0C71 1126 20$:
                                0C71 1127 RET : return
```



```
0C72 1129 .SBTTL CRE_JOB
0C72 1130 :++
0C72 1131 :FUNCTIONAL DESCRIPTION:
0C72 1132 :Routine to enter a job in queue #1
0C72 1133 :
0C72 1134 :CALLING SEQUENCE:
0C72 1135 :PUSHAL W^OPTION_LIST ; counted option list ending with a
0C72 1136 : ; byte of 0
0C72 1137 :PUSHAL W^JOB_NAME ; counted job name ending with a byte of 0
0C72 1138 :CALLS #0,W^CRE_JOB ; check buffer
0C72 1139 :
0C72 1140 :INPUT PARAMETERS:
0C72 1141 :Listed above plus initied NAMBLK to proper command file and
0C72 1142 :location MBCHAN initied to the mailbox channel.
0C72 1143 :
0C72 1144 :OUTPUT PARAMETERS:
0C72 1145 :Location JOBID contains the job ID of the created job and
0C72 1146 :the job is placed in QUE #1
0C72 1147 :
0C72 1148 :--
0C72 1149 :
0C72 1150 :CREATE: ; create a job message buffer
00000032' 0C72 1151 :.LONG CR_MSGSIZ
00000C7A' 0C76 1152 :.ADDRESS .F4
0009 0C7A 1153 :.WORD SMR$K_CREJOB
55 51 5F 54 41 42 5F 50 54 45 55 00' 0C7C 1154 :.ASCIC /UETP_BAT_QUE1/
31 45 0C88
0D 0C7C
00000C8C 0C8A 1155 :.BLKB 2
0C8C 1156 :OPTIONS:
00000CAC 0C8C 1157 :.BLKB 32
00000032 0CAC 1158 :CR_MSGSIZ=-CREATE-8
0CAC 1159 :ADDFILE: ; add a file message buffer
0000003E' 0CAC 1160 :.LONG AD_MSGSIZ
00000CB4' 0CB0 1161 :.ADDRESS .F4
000A 0CB4 1162 :.WORD SMR$K_ADDFIL
0CB6 1163 :DEVICE:
00000CC6 0CB6 1164 :.BLKB 16
0CC6 1165 :FID:
00000CCC 0CC6 1166 :.BLKB 6
0CCC 1167 :DID:
00000CD2 0CCC 1168 :.BLKB 6
4D 4F 43 2E 35 30 53 00' 0CD2 1169 :.ASCIC /S05.COM/
07 0CD2
00000CE7 0CDA 1170 :.BLKB 13
0CE7 1171 :JOBID:
0000 0CE7 1172 :.WORD 0
0CE9 1173 :JOB_NAME:
00000CF1 0CE9 1174 :.BLKB 8
00 0CF1 1175 :.BYTE 0
0000003E 0CF2 1176 :AD_MSGSIZ=-ADDFILE-8
0CF2 1177 :CLOSE: ; close a job message buffer
00000003' 0CF2 1178 :.LONG CL_MSGSIZ
00000CFA' 0CF6 1179 :.ADDRESS .F4
000B 0CFA 1180 :.WORD SMR$K_CLSJOB
00 0CFC 1181 :.BYTE 0
00000003 0CFD 1182 :CL_MSGSIZ=-CLOSE-8
```

Address	Op Code	Op Name	Comments
07FC	OCFD	1183	: CRE_JOB:
56 08 AC	OCFD	1184	WORD *M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
FF80 CF 57 86	OCFF	1186	MOVL 8(AP),R6 ; get the option list pointer
56 04 AC	OD03	1187	MOVZBL (R6)+,R7 ; get the option list size
FFD0 CF 66 57	OD06	1188	MOVCS R7,(R6),W^OPTIONS ; set the option list
FFA5 CF 03C4'CF 06	OD0C	1189	MOVL 4(AP),R6 ; get the job name pointer
FFA3 CF 03CA'CF 06	OD10	1190	MOVZBL (R6)+,R7 ; get the job name size
56 03B4'CF 56	OD13	1191	MOVCS R7,(R6),W^JOB NAME ; set the job name
FF7E CF 03B4'CF 56	OD19	1192	MOVCS #FIDSIZ,W^NAMBLK+NAMSW_FID,W^FID ; set the FID
56 00040001 8F	OD21	1193	MOVCS #DIDSIZ,W^NAMBLK+NAMSW_DID,W^DID ; set the DID
FDB8 CF 01 FB	OD29	1194	MOVZBL W^NAMBLK+NAMST_DVI,R6 ; get device name size
	OD2E	1195	INCL R6 ; include the count byte
	OD30	1196	MOVCS R6,W^NAMBLK+NAMST_DVI,W^DEVICE ; set the device name
	OD38	1197	MOVL #JBCS_NORMAL,R6 ; set expected status return
	OD3F	1198	PUSHL #0 ; set a dummy parameter
	OD41	1199	CALLS #1,W^REG_SAVE ; save a reg snapshot
	OD46	1200	\$SNSMB_S MSGBUF = W^CREATE,-
	OD46	1201	CHAN = W^MBCHAN ; create a job
	OD56	1202	FAIL_CHECK \$\$\$_NORMAL ; check for failure
00000000'8F	DD	OD56	PUSHL #\$\$\$_NORMAL
FDA7 CF 01 FB	OD5C		CALLS #1,W^REG_CHECK
OE59'CF 00 FB	OD61	1203	CALLS #0,W^SND_CHECK ; check the results
FF7A CF 017D'CF	BD	OD66	MOVW W^MBUF+2,W^JOBID ; save the job ID
		OD6D	1204
		OD6D	1205
		OD6D	1206
		OD7D	1207
		OD7D	1208
00000000'8F	DD	OD7D	1209
FD80 CF 01 FB	OD83		1210
OE59'CF 00 FB	OD88	1208	1211
		OD8D	
		OD8D	
		OD9D	
		OD9D	
00000000'8F	DD	OD9D	
FD60 CF 01 FB	ODA3		
OE59'CF 00 FB	ODA8	1212	
	ODAD	1213	
			CALLS #0,W^SND_CHECK ; check the results
			RET ; thats all folks

```
ODAE 1215 .SBTTL BUF_CHECK
ODAE 1216 :++
ODAE 1217 :FUNCTIONAL DESCRIPTION:
ODAE 1218 :Routine to check the contents of a buffer against known good
ODAE 1219 :data.
ODAE 1220 :
ODAE 1221 :CALLING SEQUENCE:
ODAE 1222 :CALLS #0,W^BUF_CHECK ; check buffer
ODAE 1223 :
ODAE 1224 :INPUT PARAMETERS:
ODAE 1225 :R6 = buffer address
ODAE 1226 :R7 = good data address
ODAE 1227 :R8 = byte count
ODAE 1228 :
ODAE 1229 :OUTPUT PARAMETERS:
ODAE 1230 :NONE
ODAE 1231 :
ODAE 1232 :--
ODAE 1233 :
ODAE 1234 BCSD:
00000050 ODAE 1235 .LONG 80
00000DB6 ODB2 1236 .ADDRESS BCBUF
ODAE 1237 BCBUF:
00000E06 ODB6 1238 .BLKB 80
OE06 1239 BCSD:
00000000 OE06 1240 .LONG 0
00000DB6 OE0A 1241 .ADDRESS BCBUF
OE0E 1242 PARAM1:
00000E1A OE0E 1243 .BLKL 3
OE1A 1244 :
OE1A 1245 BUF_CHECK:
07FC OE1A 1246 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
66 59 56 D0 OE1C 1247 MOVL R6,R9 ; save a copy of the buffer address
67 67 58 29 OE1F 1248 CMPC3 R8,(R7),(R6) ; check the buffer
5A E6 AF DE OE23 1249 BEQL 10$ ; br if good
8A 8A 63 9A OE25 1250 MOVAL B^PARAM1,R10 ; set parameter pointer
8A 8A 61 9A OE29 1251 MOVZBL (R3),(R10)+ ; save bad data
BA 53 59 C3 OE2C 1252 MOVZBL (R1),(R10)+ ; save good data
5A DB AF DE OE2F 1253 SUBL3 R9,R3,(R10)+ ; save byte offset
OE33 1254 MOVAL B^PARAM1,R10 ; reset address pointer
OE37 1255 $FAD_S CTRSTR = W^CS6,-
OE37 1256 OUTLEN = W^BCSD,-
OE37 1257 OUTBUF = W^BCSD,-
OE37 1258 P1 = (R10)+,-
OE37 1259 P2 = (R10)+,-
OE37 1260 P3 = (R10)+
FCF2 CF B3 AF DF OE50 1261 PUSHAL B^BCSD ; make the string
01 FB OE53 1262 CALLS #1,W^PRINT_FAIL ; push the string variable
04 OE58 1263 10$: ; print the failure
OE58 1264 RET ; return
```



```
00000046'8F 0171'CF D1 OE59 1266 .SBTTL SND_CHECK
               OD 13 OE59 1267 :++
               56 017D'CF B1 OE59 1268 : FUNCTIONAL DESCRIPTION:
               1C 13 OE59 1269 : Routine to check the contents of a buffer against known good
               0B 11 OE59 1270 : data.
               OE59 1271 :
               OE59 1272 : CALLING SEQUENCE:
               OE59 1273 : CALLS #0,W^SND_CHECK ; check buffer
               OE59 1274 :
               OE59 1275 : INPUT PARAMETERS:
               OE59 1276 : R6 = expected status code
               OE59 1277 :
               OE59 1278 : OUTPUT PARAMETERS:
               OE59 1279 : NONE
               OE59 1280 :
               OE59 1281 : --
               OE59 1282 :
003C OE59 1283 SND_CHECK:
               OE59 1284 : WORD ^M<R2,R3,R4,R5>
               OE5B 1285 : $QIOW_S FUNC=#10$ READVBLK,-
               OE5B 1286 : CHAN=W^MBCHAN,-
               OE5B 1287 : IOSB=W^STATUSM,-
               OE5B 1288 : P1 =W^MBUF,-
               OE5B 1289 : P2 =#80 ; read the mail
               OE82 1290 : CMPL W^SERV_NAME,#SND_SMB ; SND_SMB or SNDOPR
               OE8B 1291 : BEQL 10$ ; br if SND_SMB
               OE8D 1292 : CMPW W^MBUF+OPC$W_MS_STATUS,R6 ; correct response type?
               OE92 1293 : BEQL 30$ ; br if yes
               OE94 1294 : PUSHL W^MBUF+OPC$W_MS_STATUS ; push received
               OE98 1295 : BRB 20$ ; br to common code
               OE9A 1296 10$:
               OE9A 1297 : CMPL W^MBUF+4,R6 ; correct status return?
               OE9F 1298 : BEQL 30$ ; br if yes
               OEA1 1299 : PUSHL W^MBUF+4 ; push received
               OEAS 1300 20$:
               OEAS 1301 : PUSHL R6 ; push expected
               OEA7 1302 : PUSHAL W^EXP ; push string variable
               OEAB 1303 : CALLS #3,W^PRINT_FAIL ; print the failure
               OE80 1304 30$:
               OE80 1305 : RET
               OE81 1306 : .SBTTL GENREQ
               OE81 1307 : ++
               OE81 1308 : FUNCTIONAL DESCRIPTION:
               OE81 1309 : routine to generate a pending request for $SNDOPR
               OE81 1310 :
               OE81 1311 : CALLING SEQUENCE:
               OE81 1312 : CALLS #0,W^GENREQ ; generate a pending request
               OE81 1313 :
               OE81 1314 : INPUT PARAMETERS:
               OE81 1315 : NONE
               OE81 1316 :
               OE81 1317 : OUTPUT PARAMETERS:
               OE81 1318 : NONE
               OE81 1319 :
               OE81 1320 : --
               OE81 1321 :
               OE81 1322 : GENREQ:
```

SATSSS05
V04-000

D 10
- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 00:46:10 VAX/VMS Macro V04-00
GENREQ 5-SEP-1984 04:29:47 [UETPSY.SRC]SATSSS05.MAR;1

Page 34
(2)

00000000'BF DD
FC3A CF 01 FB
04 OECE 1327

OFFC OEB1 1323
OEB3 1324
OEB3 1325
OEC3 1326

WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
\$SNDOPR_S MSGBUF = W^OP MSG1,-
CHAN = W^MBCHAN ; generate a request
FAIL_CHECK \$\$\$_NORMAL ; check for failure
PUSHL #\$\$\$ NORMAL
CALLS #1,W^REG_CHECK
RET

```
OECF 1329 .SBTTL MODE_ID
OECF 1330 :++
OECF 1331 : FUNCTIONAL DESCRIPTION:
OECF 1332 : Subroutine to identify the mode that an exit handler is in.
OECF 1333 :
OECF 1334 : CALLING SEQUENCE:
OECF 1335 : CALLS #0,W^MODE_ID
OECF 1336 :
OECF 1337 : INPUT PARAMETERS:
OECF 1338 : MODE contains an address pointing to an ascii string desc.
OECF 1339 : of the current CPU mode.
OECF 1340 :
OECF 1341 : OUTPUT PARAMETERS:
OECF 1342 : NONE
OECF 1343 :
OECF 1344 :--
OECF 1345 :
003C OECF 1346 MODE_ID:
OECF 1347 .WORD ^M<R2,R3,R4,R5>
OED1 1348 $FAO S W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
OEEA 1349 $PUTMSG_S W^MSGVEC ; print the mode message
04 OEFB 1350 RET
```



```
05 0EFC 1352 MOD_MSG_PRINT:
    0EFC 1353 :
    0EFC 1354 :*****
    0EFC 1355 :*
    0EFC 1356 :* PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES *
    0EFC 1357 :* (USING THE PUTMSG MACRO). *
    0EFC 1358 :*
    0EFC 1359 :*****
    0EFC 1360 :
    0EFC 1361 PUTMSG <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
    OF17 1362 RSB ; ... AND RETURN TO CALLER
    OF18 1363 :
    OF18 1364 CHMRTN:
    OF18 1365 :*****
    OF18 1366 :*
    OF18 1367 :* CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER *
    OF18 1368 :* A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED *
    OF18 1369 :* BY THE MODE MACRO ('TO' OPTION). IT MERELY DOES *
    OF18 1370 :* A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS *
    OF18 1371 :* THE EFFECT OF RETURNING TO THE END OF THE MODE *
    OF18 1372 :* MACRO EXPANSION. *
    OF18 1373 :*
    OF18 1374 :*****
    OF18 1375 :
    00000059'FF 0000 0EFC 1376 .WORD 0 ; ENTRY MASK
    17 0E1A 1377 JMP @CHM_CONT ; RETURN TO MODE MACRO IN NEW MODE
    OF20 1378 :
    OF20 1379 :* RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
    OF20 1380 :
    OF20 1381 .END SATSSS05
```

SATSSS05
Symbol table

- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 00:46:10 VAX/VMS Macro V04-00
5-SEP-1984 04:29:47 [UETPSY.SRC]SATSSS05.MAR;1

Page 37
(2)

\$\$TAB	= 00000494	R	03	EXP	00000156	R	02
\$\$TABEND	= 000004E4	R	03	FAB	00000400	R	03
\$\$TMP	= 00000000			FAB\$C_BID	= 00000003		
\$\$TMP1	= 00000001			FAB\$C_BLN	= 00000050		
\$\$TMP2	= 000000CF			FAB\$C_SEQ	= 00000000		
\$\$ARGS	= 00000002			FAB\$C_VAR	= 00000002		
\$\$T1	= 00000004			FAB\$C_ALQ	= 00000010		
\$\$T2	= 00000004			FAB\$C_FOP	= 00000004		
A	= 00000064			FAB\$V_CHAN_MODE	= 00000002		
ACCSK_BATRM	= 00000002			FAB\$V_CR	= 00000001		
ACCSK_DISAACC	= 00000004			FAB\$V_FILE_MODE	= 00000004		
ACCSK_DISASEL	= 00000006			FAB\$V_LNM_MODE	= 00000000		
ACCSK_ENABACC	= 00000003			FAB\$V_PUT	= 00000000		
ACCSK_ENABSEL	= 00000005			FAB\$W_GBC	= 00000048		
ACCSK_INSMESG	= 00000001			FAB1	00000494	R	03
ACCSK_INSMMSG	= 00000011			FID	00000CC6	R	04
ACCSK_INTTRM	= 00000003			FIDSIZ	= 00000006		
ACCSK_LOGTRM	= 00000004			FILE_NAME	0000027C	R	02
ACCSK_PRCRTRM	= 00000001			FILE_NAME1	00000290	R	02
ACCSK_PRTJOB	= 00000010			FILNAMSIZ	= 00000014		
ACC_DESC	0000023E	R	03	GENREQ	00000EB1	R	04
ACC_MSG	000001E3	R	03	GET1	00000221	R	04
ACC_MSG1	00000235	R	03	GET2	00000299	R	04
ADDFILE	00000CAC	R	04	INFO	= 00000003		
AD_MSGSIZ	= 0000003E			IOSM_NOW	*****	X	04
ALC_OPR	= 00FFFF01F			IOS_READVBLK	*****	X	03
BAT_IMP_EXC	00000164	R	02	JBC\$_NORMAL	= 00040001		
BCBUF	00000DB6	R	04	JN1	000002FD	R	02
BCOSD	00000E06	R	04	JN2	00000307	R	02
BCSD	00000DAE	R	04	JN3	00000311	R	02
BUF	000000E7	R	03	JOBID	00000CE7	R	04
BUF_CHECK	00000E1A	R	04	JOB_NAME	00000CE9	R	04
BUF_SIZE	= 00000064			LF	= 0000000A		
CHMRTN	00000F18	R	04	LIB\$SIGNAL	*****	X	04
CHM_CONT	00000059	R	03	MBCHAN	00000175	R	03
CLOSE	00000CF2	R	04	MBNAM	00000140	R	02
CL_MSGSIZ	= 00000003			MBUF	0000017B	R	03
COM_FILE	00000297	R	02	MESSAGEL	00000169	R	03
COM_FIL_SIZ	= 00000007			MODE	00000177	R	03
CR	= 0000000D			MODE_ID	00000ECF	R	04
CREATE	00000C72	R	04	MOD_MSG_CODE	00000044	R	03
CRE_JOB	00000CFD	R	04	MOD_MSG_PRINT	00000EFC	R	04
CR_MSGSIZ	= 00000032			MSG\$ACCRSP	*****	X	04
CST	0000004D	R	02	MSG1C	= 00000036		
CS2	0000007F	R	02	MSG1_SIZE	= 00000009		
CS3	000000AC	R	02	MSG1	000000DF	R	03
CS5	000000DF	R	02	MSG1_VEC	000001BD	R	02
CS6	000000F4	R	02	MSG1_VEC1	000001CB	R	03
CTL\$GL_PHD	*****	X	04	MSG_LEN	= 00000080		
CURRENT_TC	00000004	R	03	MSG_SIZE	= 00000052		
DEVICE	00000CB6	R	04	NAM\$B_ESS	= 0000000A		
DIBSW_UNIT	= 0000000C			NAM\$B_NOP	= 00000008		
DID	00000CCC	R	04	NAM\$B_RSS	= 00000002		
DIDSIZ	= 00000006			NAM\$C_BID	= 00000002		
EMB\$C_SS	= 00000027			NAM\$C_BLN	= 00000060		
EMB\$W_HD_ENTRY	= 00000004			NAM\$C_ESA	= 0000000C		
ERROR	= 00000002			NAM\$C_RSA	= 00000004		

SATSSS05
Symbol table

H 10
- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 00:46:10 VAX/VMS Macro V04-00
5-SEP-1984 04:29:47 [UETPSY.SRC]SATSSS05.MAR;1

Page 38
(2)

NAMST_DVI = 00000014
NAMSW_DID = 0000002A
NAMSW_FID = 00000024
NAMBLR = 000003A0 R 03
NAME_SIZE = 00000008
OL1 = 000002F0 R 02
OL1S = 00000007
OL2 = 000002F8 R 02
OL2S = 00000004
OPCSB_MS_ENAB = 00000001
OPCSL_MS_MASK = 00000004
OPCSL_MS_OTEXT = 0000001A
OPCSL_MS_RPLYID = 00000004
OPCSL_MS_RQSTID = 00000004
OPCSL_MS_TEXT = 00000008
OPCSM_NM_CENTRL = 00000001
OPCSM_NM_DEVICE = 00000010
OPCSM_NM_DISKS = 00000008
OPCSM_NM_OPER1 = 00001000
OPCSM_NM_OPER10 = 00200000
OPCSM_NM_OPER11 = 00400000
OPCSM_NM_OPER12 = 00800000
OPCSM_NM_OPER2 = 00002000
OPCSM_NM_OPER3 = 00004000
OPCSM_NM_OPER4 = 00008000
OPCSM_NM_OPER5 = 00010000
OPCSM_NM_OPER6 = 00020000
OPCSM_NM_OPER7 = 00040000
OPCSM_NM_OPER8 = 00080000
OPCSM_NM_OPER9 = 00100000
OPCSM_NM_PRINT = 00000002
OPCSM_NM_TAPES = 00000004
OPCST_MS_ONAME = 0000000A
OPCSW_MS_OUNIT = 00000008
OPCSW_MS_STATUS = 00000002
OPCS_RQSTABORT = 0005801C
OPCS_RQSTCAN = 00058084
OPCS_RQSTCPLTE = 00058029
OPCS_RQSTPEND = 00058021
OPCS_RQ_CANCEL = 00000005
OPCS_RQ_REPLY = 00000004
OPCS_RQ_RQST = 00000003
OPCS_RQ_TERME = 00000001
OPMSG = 00000296 R 03
OPMSG_DESC = 0000028E R 03
OPNAME = 00000239 R 02
OPTIONS = 00000C8C R 04
OPTYPE = 00000246 R 03
OP_MSG = 0000024E R 02
OP_MSG_LEN = 0000002E
OP_MSG1 = 0000023E R 02
PARAM1 = 00000E0E R 04
PHDSQ_PRIVMSK = 00000000
PRINT_FAIL = 00000B4A R 04
PRIVMSK = 00000051 R 03
PRIV_ARGS = 00000002
PRVSV_BUGCHK = 00000017

PRVSV_OPER = 00000012
PRVPRT = 00000050 R 03
QIO = 00000069 R 03
QIOS_ASTADR = 00000014
QIOS_ASTPRM = 00000018
QIOS_CHAN = 00000008
QIOS_EFN = 00000004
QIOS_FUNC = 0000000C
QIOS_IOSB = 00000010
QIOS_NARGS = 0000000C
QIOS_P1 = 0000001C
QIOS_P2 = 00000020
QIOS_P3 = 00000024
QIOS_P4 = 00000028
QIOS_P5 = 0000002C
QIOS_P6 = 00000030
QUENAM1 = 000001A1 R 02
QUENAM1L = 0000000E
QUENAM2 = 000001AF R 02
QUENAM2L = 0000000E
RAB = 00000450 R 03
RABSB_RAC = 0000001E
RABSC_BID = 00000001
RABSC_BLN = 00000044
RABSC_SEQ = 00000000
RABSL_CTX = 00000018
RABSL_RBF = 00000028
RABSL_ROP = 00000004
RABSW_RSZ = 00000022
READ_CHECK = 00000C0E R 04
RECO_SIZE = 00000025
REC1 = 000002BC R 02
REC1_SIZE = 0000000B
REC2 = 000002C7 R 02
REC2_SIZE = 00000029
REG = 000000C9 R 03
REGNUM = 000000DB R 03
REG_CHECK = 00000B08 R 04
REG_SAVE = 00000AFE R 04
REG_SAVE_AREA = 00000008 R 03
RETADR = 0000005D R 03
SATSSS05 = 00000000 RG 04
SERV_NAME = 00000171 R 03
SEVERE = 00000004
SHRSK_SHRDEF = 00000001
SHRS_TEXT = 00001130
SMOSK_DETJOB = 00000043
SMOSK_DISWAP = 0000004E
SMOSK_HOLD = 00000021
SMOSK_INIPRI = 0000004D
SMOSK_JOBLIM = 0000004C
SMOSK_JOBPRI = 00000022
SMOSK_PARAMS = 00000026
SMRSK_ADDFIL = 0000000A
SMRSK_ALTER = 0000000D
SMRSK_CLSJOB = 0000000B
SMRSK_CREJOB = 00000009

SATSSS05
Symbol table

I 10
- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 00:46:10 VAX/VMS Macro V04-00
5-SEP-1984 04:29:47 [UETPSY.SRC]SATSSS05.MAR;1

Page 39
(2)

SMR\$K_DELETE	=	00000001		
SMR\$K_INITIAL	=	00000000		
SMR\$K_MERGE	=	00000004		
SMR\$K_PAUSE	=	00000003		
SMR\$K_RELEASE	=	0000000F		
SMR\$K_RMVJOB	=	0000000C		
SMR\$K_START	=	00000002		
SMR\$K_STOP	=	00000007		
SMR\$K_SYNCJOB	=	00000011		
SMSG		0000031E	R	03
SMSG1		00000330	RR	03
SMSG_DESC		00000316	R	03
SMSG_LEN	=	0000006C		
SNDA		0000009D	R	03
SNDACC		00000031	R	02
SNDACC\$_CHAN	=	00000008		
SNDACC\$_MSGBUF	=	00000004		
SNDACC\$_NARGS	=	00000002		
SNDE		000000A9	R	03
SNDERR		00000038	R	02
SNDERR\$_MSGBUF	=	00000004		
SNDERR\$_NARGS	=	00000001		
SND0		000000B1	R	03
SNDOPR		0000003F	R	02
SNDOPR\$_CHAN	=	00000008		
SNDOPR\$_MSGBUF	=	00000004		
SNDOPR\$_NARGS	=	00000002		
SNDS		000000BD	R	03
SNDSMB		00000046	R	02
SNDSMB\$_CHAN	=	00000008		
SNDSMB\$_MSGBUF	=	00000004		
SNDSMB\$_NARGS	=	00000002		
SND_CHECK		00000E59	R	04
SS\$NORMAL		*****	X	04
STATUS		000001DB	R	03
STATUSM		00000065	R	03
STEP	=	00000018		
STP0		0000003D	R	04
STP1		000000DB	R	04
STP10		000003DC	R	04
STP11		00000450	R	04
STP12		00000543	R	04
STP13		00000575	R	04
STP14		000005B9	R	04
STP15		00000653	R	04
STP16		000006A9	R	04
STP17		000006F8	R	04
STP18		00000747	R	04
STP19		000007C5	R	04
STP2		0000010C	R	04
STP20		00000813	R	04
STP21		00000848	R	04
STP22		000008CE	R	04
STP23		00000917	R	04
STP24		00000A0A	R	04
STP3		0000013D	R	04
STP4		0000016E	R	04

STP5		0000019F	R	04
STP6		0000025D	R	04
STP7		000002C6	RR	04
STP8		00000318	RR	04
STP9		00000366	R	04
ST\$V_INHIB_MSG	=	0000001C		
SUCCESS	=	00000001		
SYM		0000038A	R	03
SYM_DESC		00000195	RR	02
SYM_NAME		00000192	R	02
SY\$CLOSE		*****	GX	04
SY\$CMKRN		*****	GX	04
SY\$CONNECT		*****	GX	04
SY\$CREATE		*****	GX	04
SY\$CREMBX		*****	GX	04
SY\$DASSGN		*****	GX	04
SY\$DELLOG		*****	GX	04
SY\$DERLMB		*****	X	04
SY\$DISCONNECT		*****	GX	04
SY\$ERASE		*****	GX	04
SY\$EXIT		*****	GX	04
SY\$FAO		*****	X	04
SY\$GETCHN		*****	GX	04
SY\$HIBER		*****	GX	04
SY\$PUT		*****	GX	04
SY\$PUTMSG		*****	GX	04
SY\$QIO		*****	GX	04
SY\$QIOW		*****	GX	04
SY\$SETPRN		*****	GX	04
SY\$SETPRV		*****	GX	04
SY\$SNDACC		*****	GX	04
SY\$SNDERR		*****	GX	04
SY\$SNDOPR		*****	GX	04
SY\$SND\$SMB		*****	GX	04
SY\$TRNLOG		*****	GX	04
SY\$WAITFR		*****	GX	04
SY\$WAKE		*****	GX	04
TEST_ERROR		000001CD	R	02
TEST_MOD_BEGIN		00000019	R	02
TEST_MOD_FAIL		0000002A	RR	02
TEST_MOD_NAME		00000000	RR	02
TEST_MOD_NAME_D		00000009	RR	02
TEST_MOD_SUCC		0000001F	RR	02
TMD_ADDR		0000004C	R	03
TMN_ADDR		00000048	R	03
TPID		00000000	R	03
TTNAM		0000014F	RR	02
TTUNIT		00000154	R	02
UETPS_SATSMS	=	007480D9		
UETPS_TEXT	=	00741133		
UM		00000134	R	02
WARNING	=	00000000		
YES		0000019D	R	02
YES_DESC		0000018A	R	02

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
RODATA	0000031B (795.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	000004E4 (1252.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS05	00000F20 (3872.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.08	00:00:00.43
Command processing	152	00:00:00.73	00:00:03.13
Pass 1	589	00:00:24.73	00:00:38.18
Symbol table sort	0	00:00:02.46	00:00:02.81
Pass 2	297	00:00:05.77	00:00:07.90
Symbol table output	39	00:00:00.27	00:00:00.28
Psect synopsis output	3	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1119	00:00:34.06	00:00:52.76

The working set limit was 2000 pages.
144825 bytes (283 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1646 non-local and 19 local symbols.
1381 source lines were read in Pass 1, producing 40 object records in Pass 2.
98 pages of virtual memory were used to define 88 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SHRLIB]UETP.MLB;1	12
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	6
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	67
TOTALS (all libraries)	85

2203 GETS were required to define 85 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:SATSSS05/OBJ=OBJ\$:SATSSS05 MSRC\$:SATSSS05/UPDATE=(ENH\$:SATSSS05)+EXECML\$/LIB+SHRLIB\$:UETP/LIB

0421 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY